
# Aplicació d'esport per a dispositius iOS

Memòria del projecte final de màster

**Màster d'aplicacions multimèdia**

Màster professionalitzador

**David Montenegro Martínez**

Professor: Jose Antonio Morán

Consultor: Sergio Schvarstein Liuboschetz

1 d'octubre de 2013

*Tots els recursos emprats per l'aplicació han estat generats pel propi autor. El seu ús i distribució es limitarà a l'àmbit acadèmic, amb el consentiment d'aquest.*

*Als meus estimats pares,*

*a la meva "germaneta" Lídia,*

*als amics incondicionals...*

## Abstract

Actualment la mobilitat està jugant un paper molt important en la societat, i els dispositius mòbils estan adquirint una importància cada cop major, fent que la ubicació dels usuaris sigui irrellevant a l'hora d'utilitzar aquests tipus de productes. És per això que empreses com *Google* i *Apple*, pioneres en tecnologia, han sabut veure el potencial d'aquest mercat i han desenvolupat sistemes ajustats a aquesta filosofia.

Així doncs, aquest projecte tractarà en dur a terme el l'analisi, el disseny i la implementació d'un producte per a *smartphones*, aplicant una temàtica de caire esportiu, tant comú i accessible per a tothom com el *running*. Aquesta aplicació estarà orientada per a usuaris de sistemes operatius iOS.

L'aplicació dotarà als usuaris de la possibilitat de grabar els seus recorreguts, les rutes dutes a terme mitjançant aquest esport. Seran capaços de visualitzar el recorregut en un mapa, i obtenir-ne diferents valors associats com el temps total, la distància recorreguda, la velocitat mitjana i les calories cremades (entre d'altres). Un cop implementat el projecte, caldrà analitzar quina es la resposta dels usuaris per a saber si ha tingut èxit i/o quines millores es poden dur a terme segons el *feedback* dels usuaris.


**Paraules clau:** Treball de fi de màster, Projecte de fi de màster, LaTeX (LaTeX), Memòria, Apple, App, Sports, *Running, Workout*, Salut, Tecnologia.

## Agraïments

Els primers agraïments m'agradaria dirigir-los als caps de l'empresa on he iniciat la meva aventura en aquest sector professional, per donar-me el vistiplau i el suport per a dur a terme un projecte com aquest, i la predisposició a distribuir l'aplicació mitjançant un certificat de distribució *Ad Hoc*.

D'altra banda, agraïr el suport del consultor Sergio, per la disponibilitat aportada durant la durada del projecte, pels suggeriments proposats i per permetre el desenvolupament del mateix.

## Notacions i convencions

En aquesta memòria s'han utilitzat dos tipus diferentes de fonts:

- **Computer Modern**: Lletra per defecte de LaTeX, dissenyada per *Donald Knuth*, categoria Serif. Utilitzada als títols i al cos del document.
- **Courier New**: Nova versió de Courier, dissenyada per *Howard Kettler*, categoria Serif. Utilitzada als fragments de codi font incrustats.

# Índex

# Índex de figures

# 1 Prefaci

Avui dia la tecnologia és un dels sectors que avança més ràpid i incrementa el valor del seu mercat de forma exponencial. És per això que la quantitat de productes relacionats amb la tecnologia que podem trobar, augmenta cada dia més i més. Entre ells podem destacar l'ús dels dispositius mòbils (d'ara en endavant *smartphones*).

Els factors "temps" i "ubicació" passen a un segon pla quan es vol interactuar amb la tecnologia. Qualsevol moment i qualsevol lloc és bo per a fer-ne ús i poder gaudir de l'àmpli ventall de possibilitats que ofereixen.

Cada marca ofereix als seus usuaris tot tipus d'aplicacions mitjançant el seu respectiu repositori d'apps (p.e: *Google* ofereix *Google Play*, *Apple* ofereix l'*App Store*). En aquest projecte treballarem sobre el sistema operatiu iOS, de la marca *Apple*, així doncs sobre el repositori que s'anomena *App Store*. És una plataforma on es connecten els usuaris a fi d'obtenir-ne recursos (bàsicament codi font) en forma d'aplicació.

**Figura 1:** Logo *App Store*

Per tal que aquest repositori mantingui la reputació de la marca i un mínim de qualitat dels recursos, *Apple* ha generat un conjunt de normes i restriccions (tant de qualitat, estabilitat i contingut) que han de complir els desenvolupadors per tal que les aplicacions siguin acceptades i posteriorment disponibles al gran públic. Durant el desenvolupament d'aquest projecte, caldrà que l'aplicació compleixi totes les normes imposades.

L'elaboració d'una aplicació real (anàlisi, disseny i implementació) d'un projecte d'aquestes dimensions requereix el domini sobre diferents àmbits de coneixement (desenvolupament del codi font, familiarització amb la plataforma de desenvolupament, elaboració de recursos, ...) que fan que el projecte sigui complex i excitant a la vegada. Aquestes són dues de les raons per les quals s'ha decidit dur a terme aquest projecte, la multitud d'aspectes associats, i sobretot, la possibilitat d'oferir un producte a milers de milions de persones d'arreu del món.

# 2 Descripció del projecte

El projecte consisteix en l'anàlisi, el disseny i la implementació d'una aplicació per a disposius mòbils amb sistema operatiu iOS. El *target* és un grup d'esportistes que combinen les activitats físiques amb les noves tecnologies.

Els esportistes aficionats al *running* que disposin d'un *iPhone*, tindran d'aquí molt poc l'opció de registrar els seus recorreguts amb una nova aplicació. Aquesta aplicació els permetrà guardar les coordenades de les seves rutes i les característiques associades a la mateixa. Entenem per característiques: La distància recorreguda, la velocitat mitjana, el desnivell (tant positiu com negatiu), les calories cremades (factor que variarà en funció de l'alçada i el pes), a més d'un històric d'activitats realitzades.

**Figura 2:** Icona *Track My Run*

La idea és intentar oferir a l'usuari una navegació molt fluida i intuïtiva, tal i com s'ha vist a assignatures del curs, l'apartat del disseny de les interfícies és un pas primordial pel bon funcionament de qualsevol aplicació multimèdia.

Un cop desenvolupat el projecte i implantada l'aplicació resultant, s'haurà de veure la resposta dels usuaris i quines millores es poden aplicar.

# 3 Objectius

Els objectius d'aquest treball final de màster es poden dividir en dos tipus, els principals i els secundaris. Els objectius principals tracten sobre les metes que cal haver superat un cop finalitzat el projecte passant per totes les fases que en formen part, i els objectius secundaris poden ser una vista a baix nivell (més detallada) dels objectius a gran escala, per això poden patir petites variacions.

## 3.1 Objectius principals

Objectius des d'una perspectiva global, clarament segmentats:

- **Context**: Existeixen un gran nombre d'aplicacions similars a l'*App Store*, així que caldrà fer un balanç amb les aplicacions que hi ha actualment al mercat, i trobar un factor clau per a diferenciar-nos de la resta. És un procés complicat, però el pròpi anàlisi ens permetrà conèixer les tendències actuals i adaptar-nos a un contingut o un altre.
- **Anàlisi, disseny i implementació**: La part mes tècnica, dur a terme una bona organització de les dades, visualització de la informació i tractament de la mateixa, i fer d'aquest conjunt un pack estable, robust i eficient.
- **Implantació**: Regir l'aplicació a les normes estipulades per *Apple*, superar el procés de validació i veure la resposta dels usuaris. A partir d'aquest punt, esperar el *feedback* corresponent i dur a terme les millores que es considerin oportunes.
- **Creixement personal**: Com qualsevol projecte d'aquestes característiques, créixer personal i professionalment adquirint experiència que segur que en un futur serà profitosa.

## 3.2 Objectius secundaris

Objectius des d'una perspectiva més centrada, específica.

- **Estabilitat**: Crec que és una de les parts més importants que ha de satisfer una aplicació. Si no s'ofereix estabilitat als usuaris, mai s'aconseguirà que siguin fidels al producte que se'ls ofereix.
- **Precisió**: També des d'un punt de vista tècnic, fer que l'aplicació sigui el màxim de precisa possible farà que els usuaris creguin en ella i hi confiïn, recomanant-la a tercers i compartint-la a les xarxes socials.
- **Estètica**: Com totes les aplicacions multimèdia, ha de ser agradable a la vista i ha d'entrar pels ulls. Si la primera impressió no és bona o els estils no són els adeqüats, serà rebutjada d'immediat.

# 4 Metodologia

La implementació d'un producte d'aquestes dimensions ha de satisfer uns requisits mínims pel que fa la gestió de la informació i una millora en el temps d'execució. Per aquesta raó, se'ns ha plantejat la tasca d'investigar quina es la metodologia més bona per tal de realitzar el producte. Actualment hi ha moltes metodologies de desenvolupament de multimèdia eficients i diferenciables, passant des de la més antiga com la metodologia *Waterfall*, fins a les tècniques mes modernes com *Agile*.

En un principi, la idea és desenvolupar el projecte utilitzant una metodologia iterativa i incremental, creada en resposta a les debilitats del model tradicional de cascada. Els origens del desenvolupament àgil es remonten a l'any 2001, quan disset enginyers d'alt nivell en el camp del desenvolupament es van reunir a Utah per explorar i compartir quin consideraven que havia de ser el futur del desenvolupament de software. En aquesta reunió els integrants van constituir l'*Agile Alliance*, i van escriure *Manifesto for agile software development*, un conjunt d'estaments que han servit com a declaració de valors i principis que han de seguir les metodologies àgils.

- **Persones i interaccions** per sobre de processos i eines.
- **Programari que funciona** per sobre de documentació exhaustiva.
- **Resposta al canvi** per sobre del seguiment d'una planificació.
- **Col·laboració amb el client** per sobre de negociació de contractes.

Tot i tendir per aquesta metodologia de desenvolupament, no s'ha de deixar de banda una definició específica dels requisits funcionals, el seu disseny i el seu desenvolupament. Aquesta metodologia ens permetrà, entre d'altres coses, avançar de manera **progressiva** i **incremental**.

## 4.1 Conceptes bàsics de la metodologia

La idea principal d'aquesta metodologia consisteix en desenvolupar un sistema de manera incremental, permetent al desenvolupador l'avantatge de poder implementar el que ha après durant el procés anterior. L'avantatge prové de dues vessants: el desenvolupament del sistema i el seu ús. Els passos clau són començar amb una implementació senzilla, i progressivament millorar la seqüència fins aconseguir el sistema complet. S'entén que aquesta filosofia de desenvolupament afegeix més èmfasi a l'adaptabilitat que a la provisionalitat.

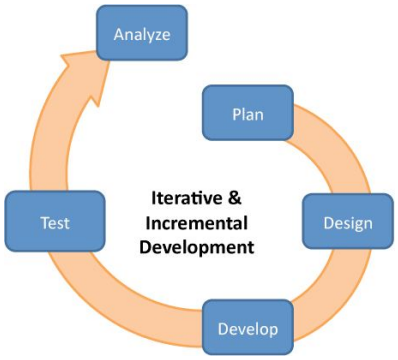


**Figura 3:** Model àgil

# 5 Planificació

Aquest apartat correspon a la planificació del projecte, entre d'altres, segmentada amb les diferents entregues de les Practiques d'Avaluació Continuada (PAC). Tal i com es pot visualitzar, el rang temporal està compost des del dia 16 d'octubre fins al 12 de gener (ambdos inclosos).

Aquesta planificació està basada en unes idees i fonaments inicials, és a dir, pot patir petites variacions durant el transcurs del projecte. Tot i així, tots i cada un dels canvis (per petits que siguin) seran documentats.

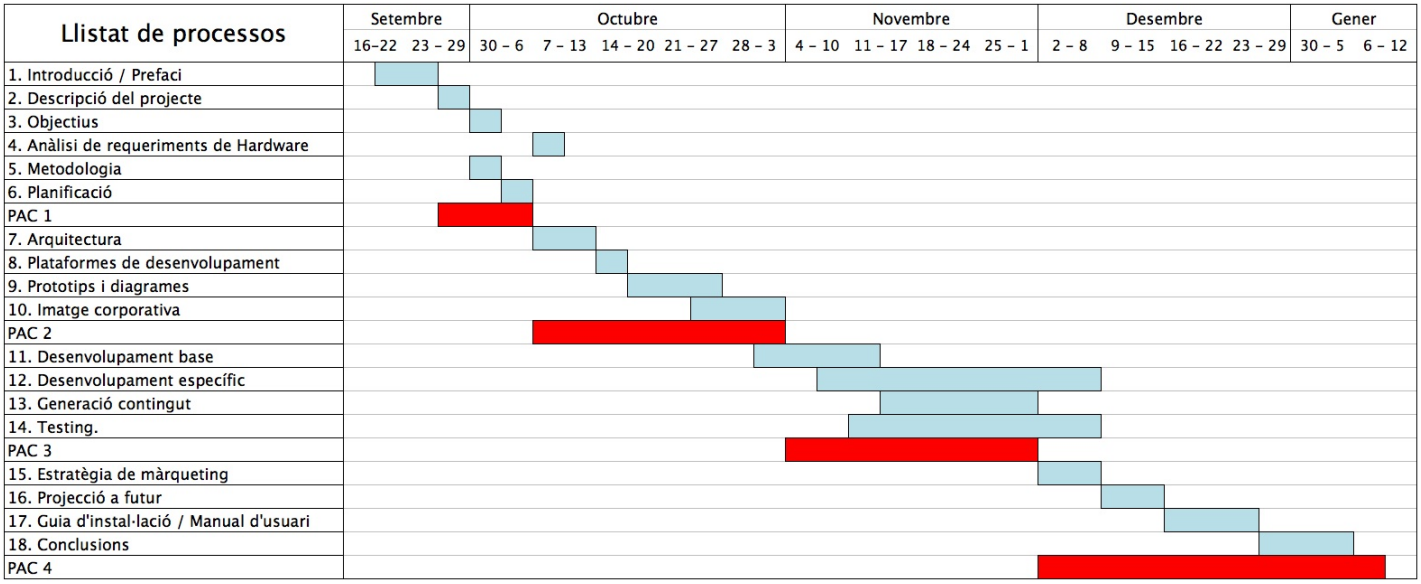


**Figura 4:** Planificació

# 6 Marc de treball i conceptes prèvis

Aquesta secció situa el projecte en el seu marc de treball, és a dir, quin és el seu origen, quins són els seus fonaments, i quin hauria de ser l'objectiu del mateix (no del projecte en sí, sinó del processos que el formen).

Així doncs, aquest projecte no ha sorgit d'una necessitat de mercat, no ha sorgit d'un buit on el mercat necessitava una funcionalitat tant específica com aquesta, sinó que ha sorgit de les ganes/expectatives d'un alumne de la UOC, específicament d'un estudiant de Màster d'Aplicacions Multimèdia, el pròpi narrador d'aquesta memòria.

**Figura 5:** Desenvolupament de plataformes per a *smartphone*

Situant-nos al Setembre de 2013, dos mesos després d'obtenir el títol d'Enginyeria Tècnica en Informàtica de Gestió proporcionat per la Universitat de Girona, vaig decidir ampliar els meus coneixements en l'àmbit multimèdia, un sector que experimenta canvis en molt poc temps i permet conèixer el ventall de possibilitats a l'hora de dur a terme una aplicació de qualsevol tipus. A més, la influència de treballar en una empresa de desenvolupament de software, més concretament en el desenvolupament d'aplicacions per a *smartphones*, va ser un dels motius que va incitar-me a cursar aquest màster.

Un cop quasi finalitzats els estudis, i haver superat la majoria d'assignatures, havia de pensar en fer un treball que aglutinés molts dels conceptes adquirits durant el curs. Així doncs, la realització d'aquest projecte ha sorgit de la necessitat de solidificar els coneixements adquirits amb el màster, havent après prèviament els conceptes de la plataforma de desenvolupament gràcies a l'oportunitat de treballar en una empresa de l'àmbit.

El fet d'executar un projecte des del seu naixament fins aconseguir els seus objectius és el que ens fa ser professionalment independents i del que es tracta en qualsevol formació: assolir uns coneixements determinats i explotar-los. Aquest treball és una mostra d'això, i té l'objectiu de superar tots els requeriments tant d'**estructuració, planificació, analítica, desenvolupament, execució i d'explotació**.

## 6.1 Anàlisi de mercat

Actualment el mercat de les aplicacions mòbils ja està molt saturat: existeixen aplicacions de tot tipus, per a qualsevol plataforma, i de qualsevol temàtica. És un mercat que ja ha omplert la majoria de forats, i moltes necessitats. Avui dia, la majoria de les aplicacions són simplement millores d'altres, o empreses que busquen promocionar-se a través d'aquestes vies.

Pel que es relaciona amb aquest projecte, cal dir que com s'ha vist prèviament, s'ha escollit desenvolupar-lo per a plataformes amb sistema operatiu iOS, però està pensat bàsicament per executar-lo sobre dispositius *iPhone*. Segons els analistes, el mercat actual situa en primera posició els dispositius *Google* amb sistema operatiu *Android*, i en segon lloc *Apple* amb sistema operatiu iOS. El següent gràfic representa la distribució actual del mercat.

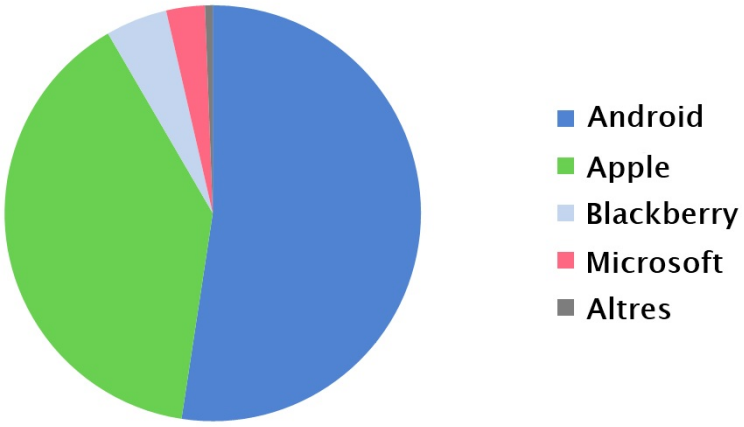


**Figura 6:** Distribució de OS, Q2, 2013

Centrant-nos ja en el nostre *target*, els usuaris d'*iPhone*, representen aproximadament el 41%, després d'*Android* que representa el 54%.

Existeixen altres aplicacions similars al projecte actual. Una de les més conegudes s'anomena *Runtastic*. Aquesta aplicació va començar essent únicament per a corredors (tal i com diu el seu nom), però més endavant va derivar a tot tipus d'esports, és a dir, van augmentar el seu objectiu un cop van veure la resposta del gran públic. El seu pla de negoci es basa en l'obtenció d'una versió PRO, així doncs de forma gratuïta només és possible obtenir l'aplicació sense la majoria de les seves funcionalitats destacades.

Una altra aplicació que és una de les favorites al mercat, és *Endomondo*. Està basada en la mateixa filosofia que *Runtastic*, utilitza el sistema de geolocalització per registrar el recorregut dels usuaris, oferint una petita xarxa social per intercanviar experiències. El pla de negoci és idèntic, oferir un mòdul de pagament on es pot obtenir l'aplicació completa pagant un tant addicional.

Degut al temps que s'exigeix pel desenvolupament del treball, l'aplicació disposarà de les funcionalitats elementals, així que si hem de ser realistes, no podrem competir amb cap d'aquestes dues. De totes maneres, tal i com veurem a la secció de treball futur, estudiarem quins possibles mòduls es podran afegir per a fer-la més enriquidora i augmentar-ne la seva competitivitat.


**Figura 7:** Marca comercial *Runtastic*


**Figura 8:** Marca comercial *Endomondo*

# 7 Requisits del sistema

Aquesta secció descriu els requisits del sistema, els quals expliquen els objectius de l'aplicació juntament amb les funcionalitats desitjades. Els requisits que deriven del projecte poden ser de dos tipus:

- **Requisits funcionals:** Descriuen quins són els serveis que ens oferirà l'aplicació, independentment de la implementació.
- **Requisits no funcionals:** Informen sobre les restriccions que venen imposades pel client o pel propi problema.

## 7.1 Requisits funcionals

S'entén com a requisits funcionals del sistema les principals funcionalitats del software a desenvolupar, representant en tot moment les responsabilitats del programa construït. En aquest apartat es descriuen totes aquelles necessitats que s'han de satisfer en termes globals, sense entrar en detall com són resoltes, ja que trobarem la resolució de problemes al llarg dels següents apartats.

Així doncs, podem desglossar els seguents requisits funcionals:

- Registrar les rutes dels corredors a través del gestor de localització del dispositiu, sempre i quan l'usuari ho permeti/desitgi.
- Visualitzar les rutes fetes pels usuaris sobre un mapa utilitzant les coordenades rebudes.
- Habilitar un perfil (local) pel corredor, amb propietats que ens permetran calcular de manera exacta el desgast calòric del mateix.
- Generar un informe d'estadístiques tals com: distància, velocitat i desnivell en el SI (Sistema Internacional d'Unitats).
- Editar certa informació relavita a les rutes: possiblement el nom o la descripció.

## 7.2 Requisits no funcionals

En tot projecte s'ha de parar especial atenció a tots els aspectes que fan referència al disseny (tant del sistema com de les interfícies), més enllà de l'explicació funcional detallada al punt anterior. Fan referència a les restriccions del tipus de disponibilitat de recursos, seguretat o interfícies externes, entre d'altres.

### 7.2.1 Requisits de software

Aquests requisits permetran executar l'aplicació sense problemes.

- L'aplicació ha de ser eficient: no s'han de fer càlculs redundants ni utilitzar més memòria de la necessària.
- Ha d'estar disponible per a qualsevol usuari amb un dispositiu iOS (orientat a usuaris amb *iPhone*).
- Extensible i modular: ha de permetre incorporar noves variants d'algoritmes a mesura que es vagi desenvolupant.
- Ha de disposar d'una interfície més que amigable, intuïtiva i usable. Vistes amb un disseny molt acurat.
- Derivat del punt anterior, generar uns recursos perfeccionistes per tal de no alterar la integritat de les vistes.
- L'aplicació haurà de disposar d'un certificat de tipus *Ad Hoc*, que permetrà que el projecte sigui avaluat per les persones corresponents, mitjançant un número únic de dispositiu (UDID).

### 7.2.2 Requisits de hardware

Per a poder dur a terme el desenvolupament del projecte, cal notar que ha sigut indispensable la disposició d'un ordinador amb sistema operatiu Mac OS X. A més, també ha sigut necessària la descàrrega del software per a poder desenvolupar l'aplicació: sense la plataforma de desenvolupament *Xcode* no hauria estat possible.

Aquest software és l'eina principal d'*Apple* per als desenvolupadors d'iOS i Mac. La nova versió del sofware (5.0) inclou vistes a les interfícies amb finestra única, suport complet per a C++ al compilador LLVM, i una integració de l'*Interface Builder* a *Xcode*. En altres seccions es descriurà amb més profunditat la plataforma de desenvolupament.

# 8 Estudis i decisions

Aquest apartat descriu tots els conceptes relacionats amb la plataforma de desenvolupament: les eines de maquinari (en el cas d'aquest projecte, específic), llibreries i el propi software (incloent el llenguatge de programació).

## 8.1 Hardware

El maquinari emprat pel desenvolupament del projecte han estat principalment un Mac OS X i un *iPhone* 4. El Mac ha estat indispensable, ja que sense un ordinador amb aquest sistema operatiu, no és possible programar nativament per a dispositius iOS (mitjançant *Xcode*).

El dispositiu mòbil *iPhone* hauria sigut prescindible, ja que la plataforma incorpora un simulador que permet igualar el comportament del dispositiu (inicialment s'havien de fer processos manuals, "*hardcoded*"pero avui dia el simulador ja ofereix quasi les mateixes prestacions que el mòbil. Un exemple és el GPS). També cal tenir en compte que el simulador s'executa utilitzant el hardware de l'ordinador, normalment amb més prestacions que el del propi dispositiu.


**Figura 9:** Propietats del Mac emprat pel desenvolupament

L'ordinador consta de les seguents prestacions:

- **Processador**: 3.4GHz Intel Core i5.
- **Memòria**: 8GB 1600 MHz DDR3.
- **Gràfics**: Gigabyte GTX 660.
- **Software**: Mac OS X Mountain Lion 10.8.5.

Les característiques del dispositiu són les següents:

- **Pantalla**: Retina Multi-Touch de 3.5 polsades (960x640).
- **Càmera**: Grabació HD (720p) i 5 megapíxels.
- **Bateria**: Recarregable de polímers de liti integrada.
- **Localització**: GPS assistit, brúixola digital, Wi-Fi.
- **Capacitat**: Unitat flash de 16GB.

## 8.2 Software

Aquesta subsecció explica les eines de programari utilitzades: entorn de programació, llenguatges, edició de diagrames, a més d'un petit raonament de la seva utilització.

### 8.2.1 Xcode

*Xcode* és l'entorn de desenvolupament integrat (IDE) d'*Apple* un complet conjunt d'eines per al desenvolupament de Mac OS X i les aplicacions d'iOS. L'IDE d'*Xcode* 5 inclou un editor de codi font de gran abast, un editor amb una sofisticada interfície gràfica d'usuari, i moltes altres característiques com un gestor de versions per a donar suport a la gestió de dipòsits de codi font. *Xcode* 5 ajuda a identificar errors tant en la sintaxi i la lògica, i fins i tot suggereix possibles solucions (igual que molts altres IDE's).

**Figura 10:** Xcode

*Xcode* 5 disposa d'una finestra de treball, que conté la major part de les dades que necessita. També hi ha una segona finestra anomenada de gestió (*Organizer Window*), i s'utilitza per organitzar els projectes i per a la lectura de la documentació. Aquesta vista disposa dels diferents projectes disponibles dins el directori, a més dels dispositius als quals es vol executar.

- **Single-View**, una sola finestra per a tots els fluxos de treball principals (pot tenir múltiples finestres i múltiples fitxes per finestra).
- **Interface Builder** és una eina gràfica totalment integrada al seu IDE, i serveis per dur a terme el disseny de les interfícies d'usuari.
- **Assistant Editor** és un panell editor de visites, el qual permet buscar i obrir fitxers relacionats amb el que s'està treballant.
- **Fix-fit** comprova els símbols i el codi de la sintaxi, destacant els errors que troba, a més de poder solucionar-los si l'usuari ho desitja.
- **Git / Subversion** són els programes amb els que treballa l'editor de versions per tal de poder mostrar tota la història d'un arxiu SMC.
- El compilador **LLVM** inclou suport complet a C, Objective-C i C++.

A l'annex de la memòria es defineixen les diferents propietats (des d'un punt de vista més tècnic).

### 8.2.2 Objective-C

El llenguatge de programació *Objective-C* està dissenyat per permetre una sofitsicada programació orientada a objectes. Aquest llenguatge es defineix com un conjunt petit però poderós d'extensions per al llenguatge estàndard ANSI C. Les seves addicions al C es basen principalment en *Smalltalk*, un dels primers llenguatges de programació orientat a objectes. La majoria dels entorns orientats a objectes consten de diverses parts:

- Un llenguatge de programació orientat a objectes.
- Una llibreria d'objectes.
- Un conjunt d'eines de desenvolupament.
- Un entorn d'execució.

### 8.2.3 LaTeX

LaTeX és un sistema de preparació de documents. Permet crear documents amb un aspecte completament professional, ja que la seva idea se centra en el contingut del document, i no la forma. Per aconseguir aquesta fita, LaTeX ofereix una sèrie de macros i estils predefinits. Com a narrador de qualsevol contingut, és important centrar-se en el contingut del document, el "*què*", mentre que el sistema s'encarrega de fer el "*com*". Les possibles desavantatges d'usar aquest programa solen ser la dificultat de la sintaxis (que analitzarem a la secció d'annexos), la visualització del document no es mostra en temps real, és a dir, ha de superar un procés de compilació. El resultat acaba essent

un document impecable (des de la meva perspectiva), sense errors de format i centrat explícitament en el contingut.

## 8.3 Llibreries

Les macros proporcionen les interfícies necessàries per escriure software. Un *framework* és un directori jeràrquic que encapsula els recursos compartits, tals com una biblioteca dinàmica compartida, imatges, seqüències localitzades, fitxers de declaració i la documentació de referència en un sol paquet.

Un *framework* és un paquet, i el seu contingut només es pot accedir usant *CoreBundle Services* de la classe *NSBundle*. De totes maneres, a diferència de la majoria de paquets, no aparèix a una carpeta com un fitxer opac, sinó com un directori estàndard pel qual s'hi pot navegar. Aquest apartat descriu doncs, els paquets utilitzats durant la implementació del projecte.

### 8.3.1 Cocoa

*Cocoa* i *Cocoa Touch* són definits com dos *frameworks* que juntament amb Mac OS X estan integrats amb l'experiència del desenvolupament amb *Xcode*. Proporciona un alt nivell d'API que fan que sigui relativament fàcil afegir una animació, la creació de xarxes, l'aparició d'una plataforma nativa i el comportament de l'aplicació en general.

**Figura 11:** Cocoa Logo

Consisteix en biblioteques, API i temps d'execució que formen la capa de desenvolupament per a tots els Mac OS X. Amb el desenvolupament de *Cocoa* es creen aplicacions de la mateixa manera que Mac OS X, ja que l'aplicació heredarà automàticament el seu comportament.

### 8.3.2 CoreLocation

Aquest és el nom que rep la llibreria encarregada de determinar la ubicació dels dispositius. Utilitza el hardware disponible per determinar la posició actual de l'usuari i la

seva direcció. Pot utilitzar les classes i els protocols per configurar i programar l'entrega de la ubicació i els esdeveniments.

**Figura 12:** CoreLocation Framework

Per establir els paràmetres que determinen cada quan s'han de cridar els esdeveniments d'ubicació i denominació, i per iniciar o finalitzar l'entrega d'esdeveniments, s'usa un objecte anomenat *CLLocationManager*, que tal i com defineix el seu nom, és la classe gestora de la localització del dispositiu. Ofereix suport per a les següents funcionalitats:

- Seguiment dels canvis de posició de l'usuari segons un grau de precisió configurable.
- Informar sobre els canvis de direcció de la brúixola.
- Seguiment de les diferents regions d'interès i la generació d'esdeveniments de la situació quan l'usuari entra o surt de regions.
- Alguns serveis de localització requereixen la presència de hardware específic per al dispositiu usat.

La utilització d'aquesta llibreria és fonamental pels objectius del projecte i satisfer els requisits del *target*. La seva funció segurament s'executarà en mode *background*.

### 8.3.3 MKMapView

Aquesta llibreria proporciona una interfície de mapes similar a la proporcionada per l'aplicació de mapes nativa. Aquesta llibreria s'utilitza per mostrar la informació relativa a les rutes dutes a terme pels usuaris, i per manipular el contingut ja sigui amb la traçada d'una línia o per situar punts (coordenades) geolocalitzades. Està clar que sense aquesta llibreria, l'aplicació perdria un dels pilars bàsics, com és la funcionalitat de mostrar el recorregut sobreposat a un mapa.

### 8.3.4 SQLiteManager

*SQLiteManager* és una classe contenidor, que disposa de mètodes per facilitar la connexió amb la base de dades de l'SDK d'iOS. Proporciona mètodes per:

- Connectar/crear una base de dades a la carpeta de documents de l'aplicació.
- Fer una consulta a la base de dades.
- Obtenir els registres en format *NSDictionary*.
- Tancar la connexió amb la base de dades.
- Exportar les dades en format SQL.


**Figura 13:** Logo SQLite

Aquesta classe ha facilitat la sintaxi que ofereix l'SDK, ja que suposa una mica complexa per les operacions que es volen dur a terme. Aquesta classe (com l'objectiu de qualsevol llibreria) té com a finalitat facilitar la interacció amb operacions simples, com en aquest cas, la interacció amb les bases de dades. També comentar que aquesta classe s'ha obtingut a través d'un repositori en línia anomenat *github*, i ha estat desenvolupada per *Anthony Ly*.

# 9 Anàlisi i disseny del sistema

En aquesta secció s'estudien i es defineixen els requisits del sistema, és a dir, el seu anàlisi, el seu disseny i la solució aplicada als requisits inicials. Per a complir aquest objectiu, s'utilitzarà el llenguatge UML ("*Unified Modeling Language*"), que és un llenguatge de modelatge estàndard dins el camp de l'enginyeria del programari. Tal i com descriu la metodologia del projecte, s'ha establert un disseny preeliminar i per a cada funcionalitat, s'aplica un disseny i una implementació.

## 9.1 Diagrames d'activitat

Es descriu el procés detallat en forma de diagrama d'activitat de la funcionalitat principal de l'aplicació. La grabació de les rutes amb les accions tant per la banda de l'usuari com la del sistema. El diagrama resultant és el següent.

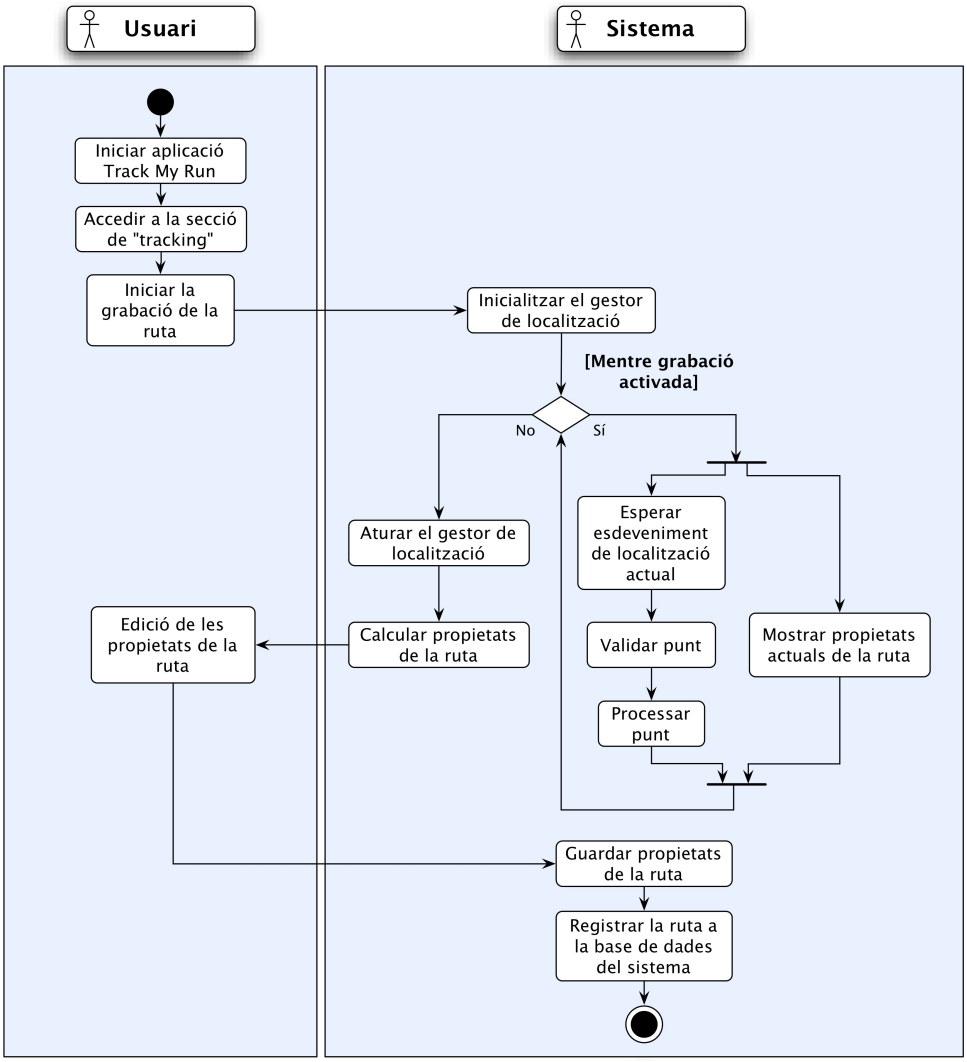


**Figura 14:** Diagrama d'activitat: Grabació ruta

Un altre dels processos més importants que realitza el sistema és el càlcul de propietats relatives a la ruta. El procés es porta a terme un cop l'usuari finalitza la grabació d'una ruta. El diagrama d'activitat és el següent:

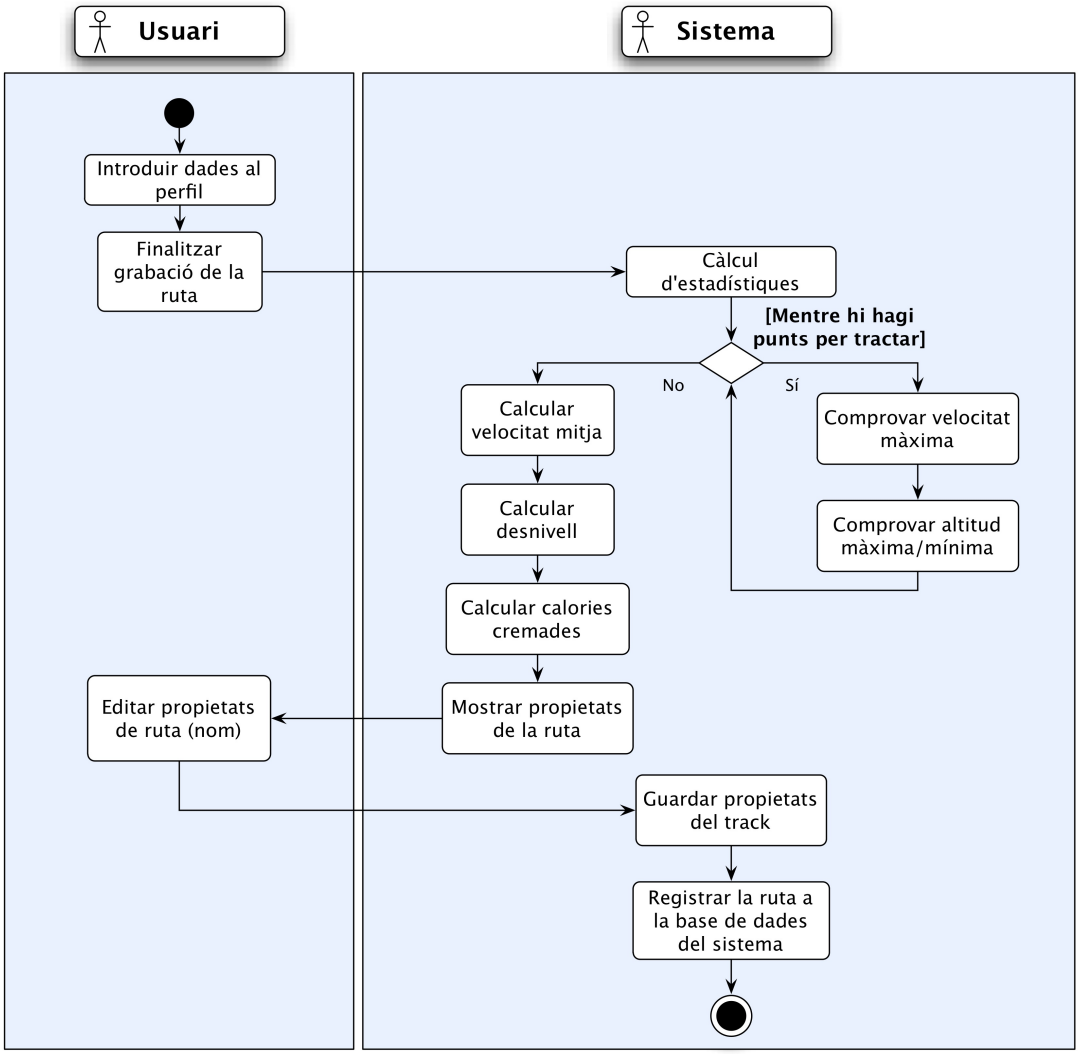


**Figura 15:** Diagrama d'activitat: Càlcul de les propietats d'una ruta

Tal i com reflexa l'últim diagrama incrustat, cal que l'usuari primer de tot introdueixi les seves dades al perfil. Llavors, s'analitzen tots els punts que han estat registrats per la ruta. Cal notar que és totalment necessàri fer una cerca pels punts rebuts per tal de diferenciar la velocitat màxima i el desnivell. Un cop fet el recorregut, l'algoritme és capaç de definir les propietats: la velocitat mitjana (a partir de la suma de la velocitat dels punts dividida pel nombre de punts), el desnivell (diferència del punt més alt amb el punt més baix), i les calories cremades (següent diagrama d'activitat). Un cop calculades, es donen d'alta a la base de dades.

El següent diagrama reflexa els càlculs que es duen a terme per obtenir el total de calories cremades:

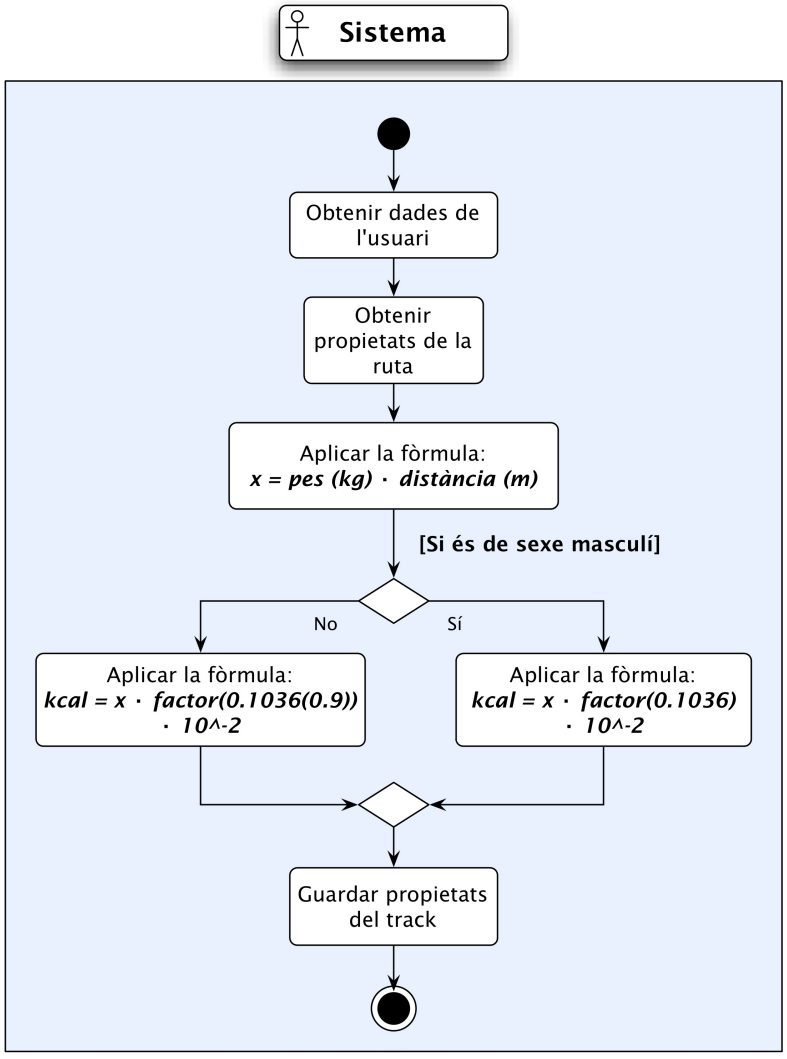


**Figura 16:** Diagrama d'activitat: Càlcul de les calories

Després de consultar a la xarxa quin és el mètode més eficient/exacte per obtenir les calories cremades fent esport (més específicament, corrent), s'ha arribat a la fòrmula aplicada. La forma més precisa d'obtenir les calories és a través del pes corporal, ja que l'alçada no deixa de ser un atribut del pes que no afecta directament les calories. L'edat en canvi, afecta principalment a la cadència del ritme cardíac, i en conseqüència a la intensitat de l'exercici, però no acostuma a ser un valor que es té en compte a les calories cremades. Així doncs, el pes és l'atribut que el nostre cos ha de desplaçar al moure's, ja que amb major pes corporal més desgast metabòlic basal. D'aquí se'n deriva el factor de desgast (0.1036) on en el cas de les dones actua en només un 90% de la seva totalitat. Un cop calculat el valor, cal multiplicar-lo pel nombre de metres fets. El resultat de l'operació per calcular les calories cremades és genèric, una referència estimada del valor real.

En el diagrama d'estadístiques que es mostra a continuació com bé diu el seu nom, genera les estadístiques. En aquest cas, només intervé el sistema, únic encarregat de generar-les:

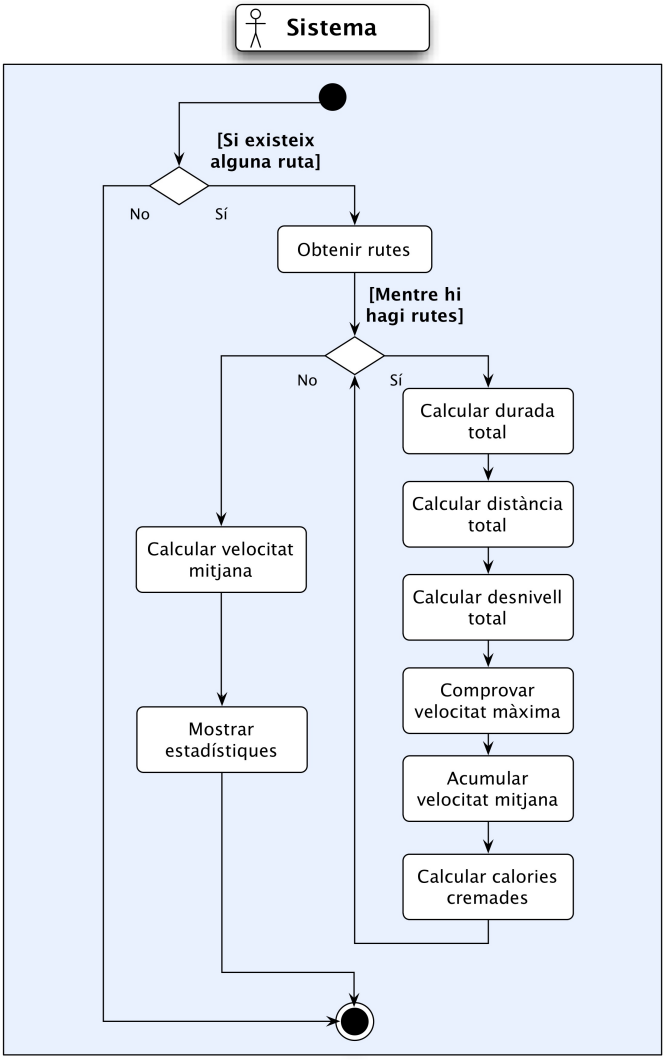


**Figura 17:** Diagrama d'activitat: Càlcul d'estadístiques

Com es pot veure, en cas que hi hagi rutes registrades, la idea és iterar per totes elles obtenint dades com la durada, la distància, el desnivell i les calories, i sumar-les fins a recòrrer-les totes. És llavors quan podrem obtenir la velocitat mitjana total, havent-les sumat totes i dividir-les pel nombre de rutes registrades. Un cop dut a terme el procés, es mostren per pantalla.

## 9.2 Model de dades

El model de dades representa de quina manera es guarda la informació a la base de dades. Mostra les diferents estructures de dades i la manera com aquestes es relacionen. La següent figura representa el model de dades del projecte.

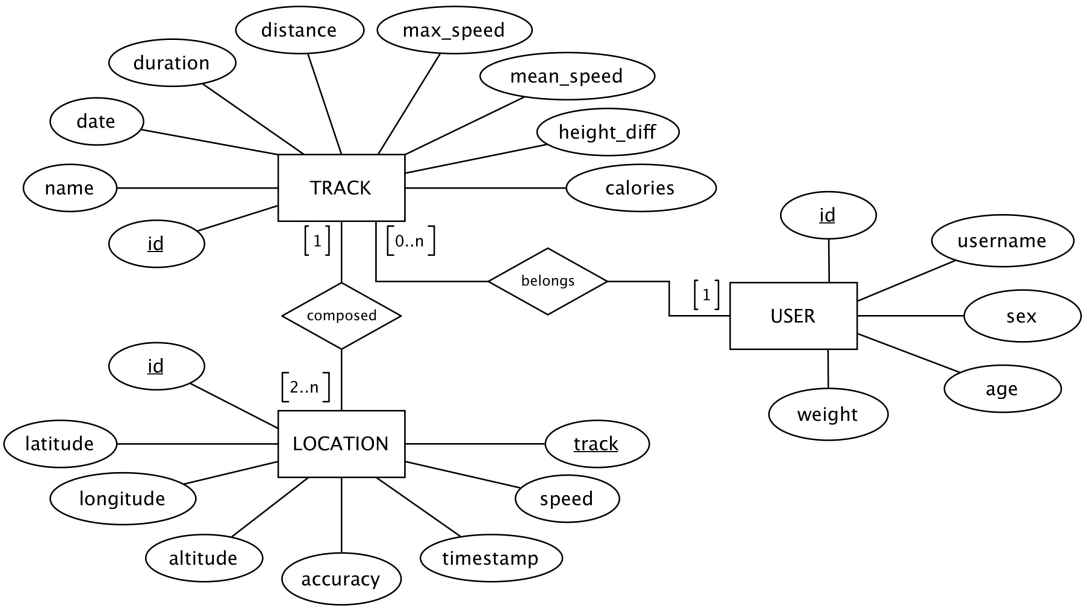


**Figura 18:** Model de dades

Cada entitat té una sèrie d'atributs que defineixen les seves propietats. Aquestes entitats es relacionen amb d'altres depenent de la manera com interactuen dins el sistema.

- **Track**: Aquesta taula defineix les estadístiques de la ruta. Un *track* està compost per com a mínim 2 punts, i està associat a un usuari en concret. Els seus atributs són: l'identificador, el nom, la data, la durada, la distància, la velocitat màxima, la velocitat mitjana, el desnivell, i el total de calories cremades.
- **Point**: El punt guarda les propietats relatives a la localització rebuda a través del dispositiu. El punt ha d'estar associat al *track* al qual formarà part. Els seus atributs són: l'identificador, la latitud, la longitud, l'altitud, la precisió, el temps, la velocitat.
- **User**: La taula usuari defineix les seguents propietats: identificador, nom d'usuari, sexe, edat i pes. Aquesta taula té un enfocament local, així doncs no podrà haver-hi múltiples usuaris.

## 9.3 Diagrama de classes

El diagrama de classes és un tipus de diagrama estàtic que descriu l'estructura del sistema, mostrant-ne les seves classes, els atributs de cada una de les classes, les seves operacions, i relacions entre elles. Aquest diagrama és utilitzat durant el procés d'anàlisi i disseny dels sistemes. Dins el diagrama, s'han segmentat les classes per diferents tipologies: classes/llibreries del sistema, classes de dades i classes de navegació.

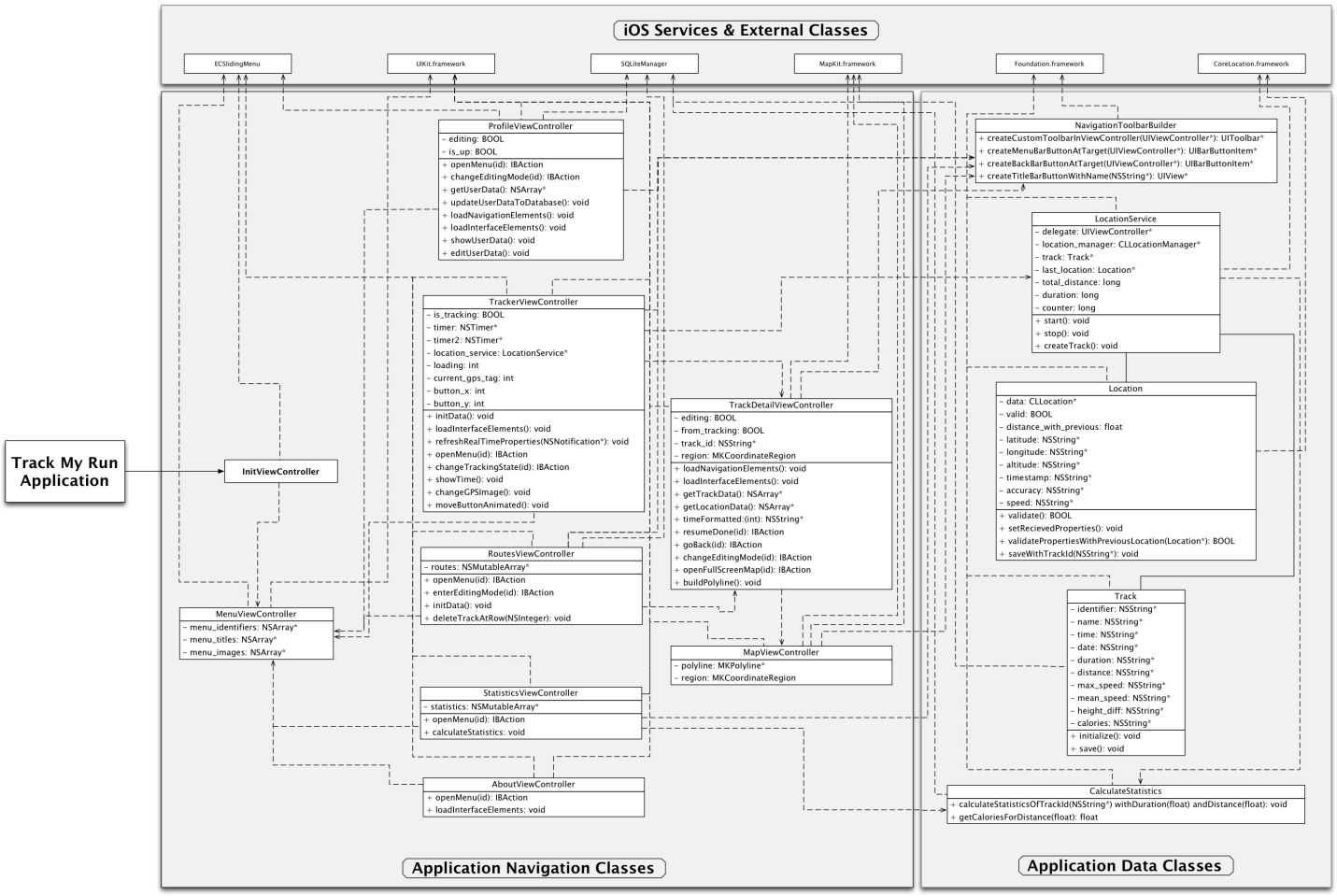


**Figura 19:** Diagrama de classes

## 9.4 Patrons de disseny

Els patrons de disseny proporcionen solucions predeterminades a problemes concrets en l'enginyeria del software. Existeixen molts tipus de patrons, uns més fàcils d'utilitzar, uns més complexos, però si s'apliquen com cal, proporcionen la solució més viable als problemes sorgits. Al llarg del projecte, s'han utilitzat patrons que han solucionat aquests problemes i s'han intentat adaptar de la millor manera possible.

**Figura 20:** Patrons de disseny

Els diferents patrons utilitzats són:

- **Delegate**: És un patró de disseny que significa "delegació", i és una tècnica en la qual un objecte expressa un cert comportament de cara a l'exterior, però en realitat delega la responsabilitat d'implementar tal comportament a un objecte associat en una relació inversa de responsabilitat.
- **Model-View-Controller**: És un patró d'arquitectura del software separa les dades d'una aplicació, la interfície d'usuari, i la lògica del negoci. El model sol ser el sistema gestor de base de dades i la lògica de negoci, i el controlador el responsable de rebre els esdeveniments d'entrada des de la vista.
- **Singleton**: És possiblement un dels patrons més senzills d'implementar. La seva funció bàsica consisteix en garantir que una classe tingui només una instància en tot el sistema, i que sigui capaç de proporcionar un punt d'accés global a aquesta instància.
- **Observer**: Aquest patró té com a context una sèrie d'objectes "observadors", i un objecte observat. Els observadors reben una notificació quan detecten que l'estat de l'objecte observat ha canviat. Es pot categoritzar com un patró de comportament, és a dir, està relacionat amb algoritmes de funcionament i assignació de responsabilitats a classes i objectes. Els patrons defineixen una classe abstracta que descriuen l'encapsulació del mateix.

# 10 Implementació i proves

Aquesta secció tracta sobre els passos que s'han seguit a l'hora de realitzar la implementació/desenvolupament del projecte. Per entendre la seva implementació, cal parlar de certs aspectes tècnics a baix nivell, detalls importants sobre les funcionalitats bàsiques i algun que altra exemple de codi. Recordar que la implementació aplicada en aquesta secció representa de forma fidel l'anàlisi descrit a l'apartat anterior.

## 10.1 Implementació dels patrons de disseny

Com s'ha dit a la secció anterior, els patrons de disseny pretenen proporcionar catàlegs d'elements reutilitzables en el disseny de sistemes de software, doncs la implementació de cada un variarà en funció de les seves necessitats: Es tracta de determinar quina problemàtica solucionen els patrons aplicats i de quina manera s'implementen.

### 10.1.1 Patró delegate

En el desenvolupament en aplicacions per iOS, s'utilitza amb gran freqüència aquest patró. L'idea principal és que una classe conté la delegació d'altres, així que declara un o més mètodes que constitueixen un protocol formal o informal. Vàries classes que s'han utilitzat per al desenvolupament del projecte utilitzen delegacions, normalment utilitzades en llibreries.

L'exemple més clar el trobem a l'hora d'iniciar el gestor de localització del dispositiu per tal d'obtenir-ne les coordenades GPS. El següent fragment de codi permet percebre amb claredat de quina manera funciona.

**Implementació del patró Delegate**

```
1  // Location Service, fitxer de declaracio
2  # import <CoreLocation/CoreLocation.h>
3  @interface LocationService : NSObject <CLLocationManagerDelegate> {
4      /* Atributs i metodes */
5  }
6
7  // Location Service, fitxer d'implementacio
8  - (void)locationManager:(CLLocationManager*)manager didUpdateToLocation:(
       CLLocation*)newLocation fromLocation:(CLLocation*)oldLocation {
9      /* Tractament de punts rebuts */
10 }
11
12 - (void)locationManager:(CLLocationManager*)manager didFailWithError:(NSError
       *)error {
13     /* Tractament d'errors */
14 }
```

### 10.1.2 Patró Model-View-Controller

*Cocoa* utilitza el patró Model-View-Controller a través del seu disseny. Els models encapsulen les dades de l'aplicació, i permeten visualitzar les vistes i editar les dades, i els controladores permeten distribuir la lògica de negoci en tres. Al separar les responsabilitats d'aquesta manera, s'acaba obtenint una aplicació que és més fàcil de dissenyar, implementar i mantenir.

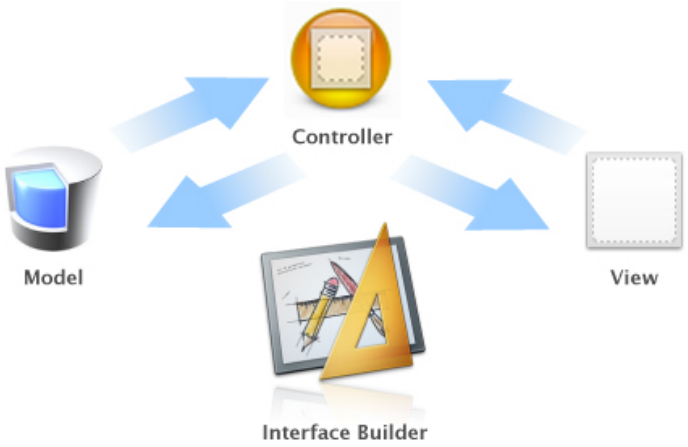


**Figura 21:** Patró Model-View-Controller

El patró descrit a alt nivell seria entès des de tres nivells: les accions que duu a terme l'usuari a través de la vista del dispositiu, el pas intermig en el que el sistema comunica la pantalla amb la gestió de dades, i la gestió de les dades a baix nivell (tant alta, baixa com modificació de les mateixes). La figura anterior representa el clar comportament del patró.

### 10.1.3 Patró Singleton

Com bé s'ha explicat anteriorment, s'encarrega de restringir la creació d'un objecte en una sola instància. El patró és dels més senzills, i el següent fragment de codi deixa clar de quina manera cal implementar-lo.

**Implementació del patró Singleton**

```
1  +(SQLiteManager*)singleton{
2      @synchronized([SQLiteManager class]) {
3        // Si l'objecte global no existeix, es crea
4        if (!sharedSQLiteManager){
5              [[self alloc] init];
6          }
7        return sharedSQLiteManager;
8     }
9     return nil;
10 }
```

### 10.1.4 Patró Observer

El patró Observer es un patró de comportament, i sol definir una relació d'un objecte a molts. Quan l'estat d'un objecte de l'aplicació canvia freqüentment, aquest patró defineix un objecte que encapsula l'aspecte, de manera que quan l'objecte canvia els que l'observen reben una notificació. És un concepte senzill d'entendre, ja que en aquest cas cada vegada que l'aplicació rebi un punt es cridarà un esdeveniment d'aquest tipus. Podem associar de forma abstracta que el mòbil rebrà un esdeveniment cada vegada que aquest canviï de posició. El següent fragment de codi permet fer-se una idea del seu funcionament a baix nivell:

**Implementació del patró Observer**

```
1 // Patro Observer
2 // S'avisa a la vista actual que quan es cridi un esdeveniment amb nom "
      tracking", cridi un fragment de codi especific
3 [[ NSNotificationCenter defaultCenter] addObserver:self selector:@selector(
      refreshRealTimeProperties:) name:@"tracking" object:nil];
4
5 // A qualsevol altre classe dins l'aplicacio
6 [[NSNotificationCenter defaultCenter] postNotificationName: @"tracking"
      object:nil userInfo:userInfo];
7 // S'envia una notificacio amb nom "tracking", i s'envia la informacio
      associada a l'esdeveniment generat
```

## 10.2 Gestor de localització

Com ja se sap, l'enregistrament de rutes es basa en rebre la senyal GPS del dispositiu. Per poder usar el GPS, cal importar la llibreria "*CoreLocation*" i implementar les funcions a la classe delegada (per saber on s'usa, mirar el diagrama de classes de l'aplicatiu). Aquesta llibreria permet al dispositiu rebre els esdeveniments de localització. El gestor s'inicialitza amb una sèrie de paràmetres específics en funció del que es vulgui obtenir. Els paràmetres específics assignats per l'aplicació són:

- **Precisió**: La màxima precisió possible del dispositiu. És un paràmetre que sol estar associat a precisió de localització (del punt original al radi de possibilitats), doncs el gestor ens avisarà amb la major precisió possible. Això no significa que sempre sigui una precisió òptima, ja que quan la senyal GPS no tingui suficient cobertura rebrà la màxima precisió possible en el moment.
- **Distància**: S'assigna la distància en la que es vol rebre la notificació. Per una banda, la precisió és sempre la millor possible, i la distància en la que volem registrar un punt s'ha optat per què sigui el valor d'un metre. És a dir, cada vegada que el dispositiu variï la seva posició un metre, es notificarà a la classe delegada del gestor.

### 10.2.1 Filtratge de localitzacions

Un cop es reben els canvis de localització del dispositiu amb les propietats esmentades en l'apartat anterior, cal processar-los a fi d'eliminar els que no compleixin una sèrie de premisses. Per tal de precisar la exactitud de cada localització rebuda, es comproven una sèrie d'atributs de la mateixa per comprovar que compleixi uns requeriments mínims.

- **Latitud i longitud**: El primer que es comprova és que les coordenades no siguin (0, 0). Mitjançant el simulador d'*iPhone* es poden generar unes localitzacions "aleatòries" per comprovar el funcionament del dispositiu, i aquestes coordenades solen ser bastant precises. En canvi, mitjançant un dispositiu real es poden rebre coordenades no vàlides, on en molts casos aquestes són (0, 0). El punt amb latitud 0 i longitud 0 se situa a l'origen del sistema de coordenades, una ubicació enmig del mar a l'est de l'Àfrica central, un lloc que mai serà un punt vàlid.
- **Precisió**: Cada localització rebuda s'atribueix a un paràmetre precisió, i aquest defineix quin és el radi d'error del punt. És a dir, la senyal GPS assegura que la ubicació real del dispositiu en una zona circular. Al iniciar el gestor, és possible que la localització sigui poc precisa, però de mica en mica la senyal sol millorar. Aquest filtre doncs, per tal de rebutjar punts erronis s'ha limitat a 40 metres, variable més que permissiva.
- **Velocitat**: També es rep un paràmetre que defineix la velocitat en cada notificació. En cas que la velocitat superi el límit establert de 37 km/h, es rebutjarà el punt. Cal recordar que aquesta és la velocitat teòrica, pot patir moltes variacions i ser poc exacta. Per tal de precisar-ne el valor, calcularem la diferència de temps entre la última i la penúltima ubicació rebuda, i s'aplicarà la següent fòrmula:

$$\mathbf{v = d\ (m)\ /\ t\ (s)}$$

- **Distància**: Per últim, es guardarà la velocitat a través de la fòrmula descrita, només en cas que la distància entre l'últim punt i el penúltim sigui inferior a 30 metres. És una variable bastant ajustada, ja que cal tenir en compte que la senyal GPS pot patir certes variacions que pot ser que interfereixin amb certs atributs de la senyal.

## 10.3 Idiomes

S'ha optat per desenvolupar l'aplicació en tres llenguatges diferents. L'anglès, el més internacional de tots, el castellà i el català degut a la regió on ens trobem. Per tal d'abarcar més regions caldria traduir-la a tots els idiomes que ofereix la plataforma d'*Apple*.

**Exemple d'implementació d'idiomes**

```
1 // Implementacio idiomes
2 [label setText:NSLocalizedString(@"LABEL_ID", nil)];
```

Amb aquest simple exemple podem observar de quina manera funcionen les traduccions. A una etiqueta determinada (associada a una vista través de l'assistent de l'*Xcode*), se li assigna un text segons la localització. La funció *LocalizedString* accedeix al fitxer de l'idioma del dispositiu, amb la clau de color vermell com a paràmetre, i n'obté el valor. És tant simple com tenir tres tipus de fitxer diferents, un per l'anglès, un pel castellà i un pel català. L'única condició és que, tots els idiomes han de tenir els identificadors que es criden, ja que en cas contrari es mostrarà el nom de la clau i és estèticament inacceptable i, lògicament, sense traduir.

# 11 Resultats visuals de l'aplicació

Aquesta secció descriu el resultat del projecte de forma majoritàriament visual, especificant per totes i cada una de les vistes quines són les seves funcionalitats principals.

Començant a descriure l'aplicació doncs, un cop es clica l'icona des des l'escriptori de l'*iPhone*, es visualitza l'anomenada pantalla de llançament, i és el moment en què l'aplicació carrega les seves dades inicials. La primera pantalla que apareix és el perfil, però el menú consta de vàries opcions. El perfil (pantalla on aterra l'aplicació), el "tracker" que ens permetrà grabar els camins, l'apartat rutes que mostra l'historial de tracks registrats, una vista d'estadístiques globals i per últim informació sobre l'aplicació i el projecte des del qual ha estat impulsada.

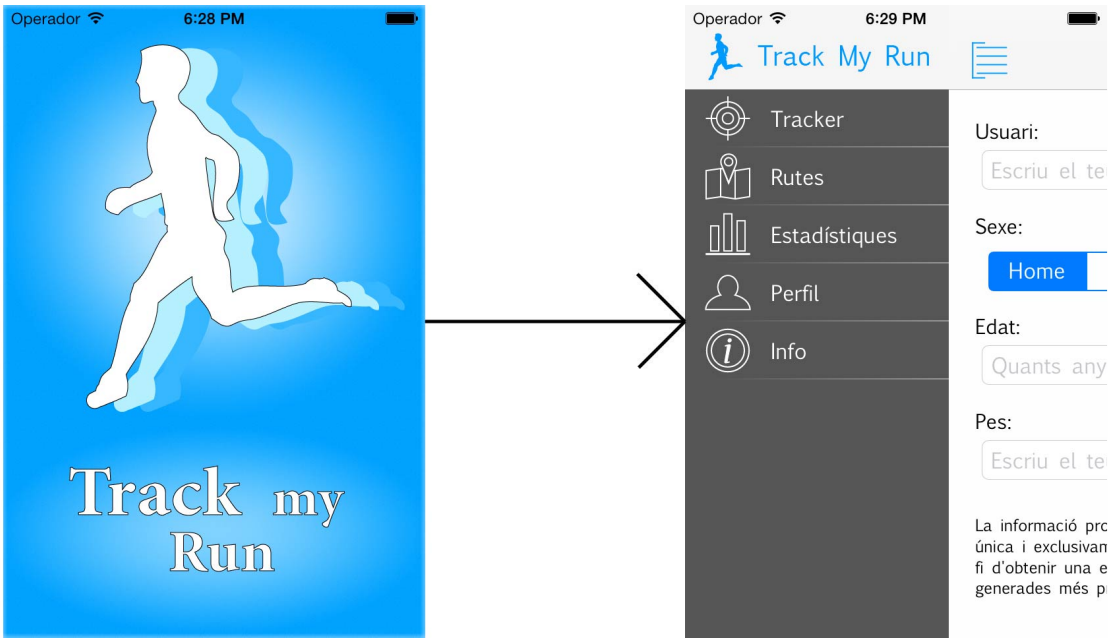


**Figura 22:** Captura de pantalla del menú

Analitzant la part de vista que es veu obrint el menú, és el perfil. És una vista que pot semblar prescindible, però gràcies a les dades que permet omplir, s'efectuarà un càlcul d'estadístiques més precís (concretament de les calories cremades). Com es veu a la següent figura incrustada, permet determinar atributs de l'usuari tals com el nom, el sexe, l'edat i el pes. Tal i com esmenta el text inferior de la vista, informa que les dades proporcionades a l'aplicació es tractaran única i exclusivament de forma local, amb la única intenció de generar un càlcul estadístic més precís.

Si es volen entrar dades, fent clic al llapis de la barra superior s'entra en mode edició. És llavors quan es mostra el teclat per omplir els camps. En cas de voler introduir l'edat i el pes, es mostra només el teclat numèric. La figura 23 representa la descripció del paràgraf feta de la vista de perfil, on es visualitza el perfil en mode normal i el perfil en mode d'edició.

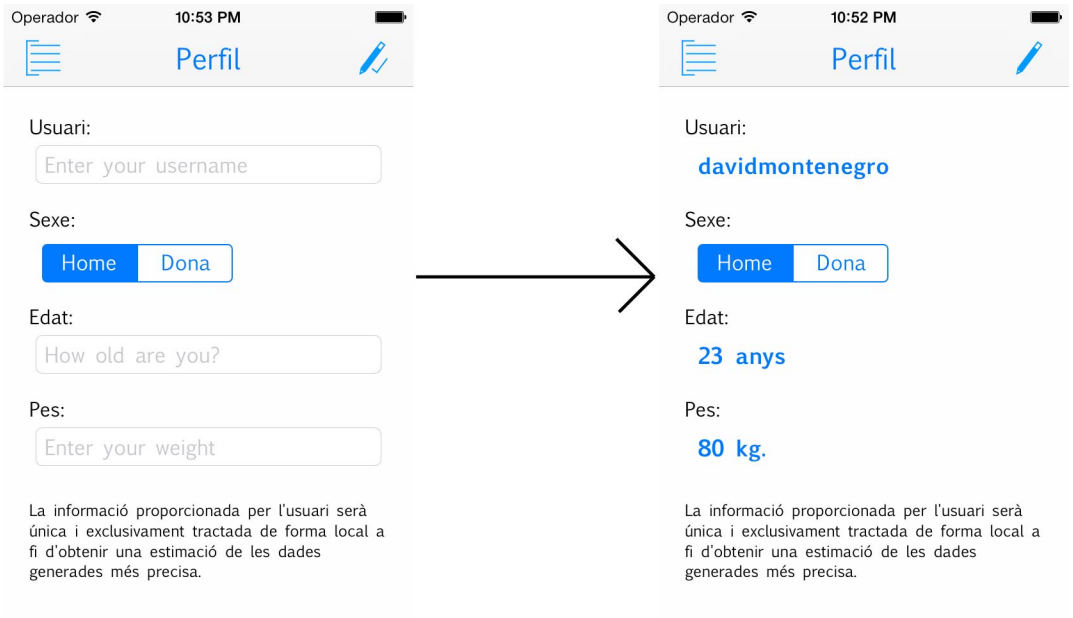


**Figura 23:** Captura de pantalla del perfil

La següent funcionalitat és probablement la més important i complexa. És la de registrar la ubicació de l'usuari en tot moment, i mostrar les dades en temps real relacionades amb el temps, la distància i la velocitat. És en aquest moment quan es duu a terme el sistema de filtratge dels punts descrit en l'apartat de desenvolupament, a fi de generar un millor traçat de la ruta. Es mostra una pantalla dividida, una amb les propietats i l'altra amb el recorregut sobre un mapa, amb l'opció de visualitzar-lo en pantalla completa. L'últim pas per finalitzar la grabació d'un track és escrivint el nom per distingir-lo de la resta.

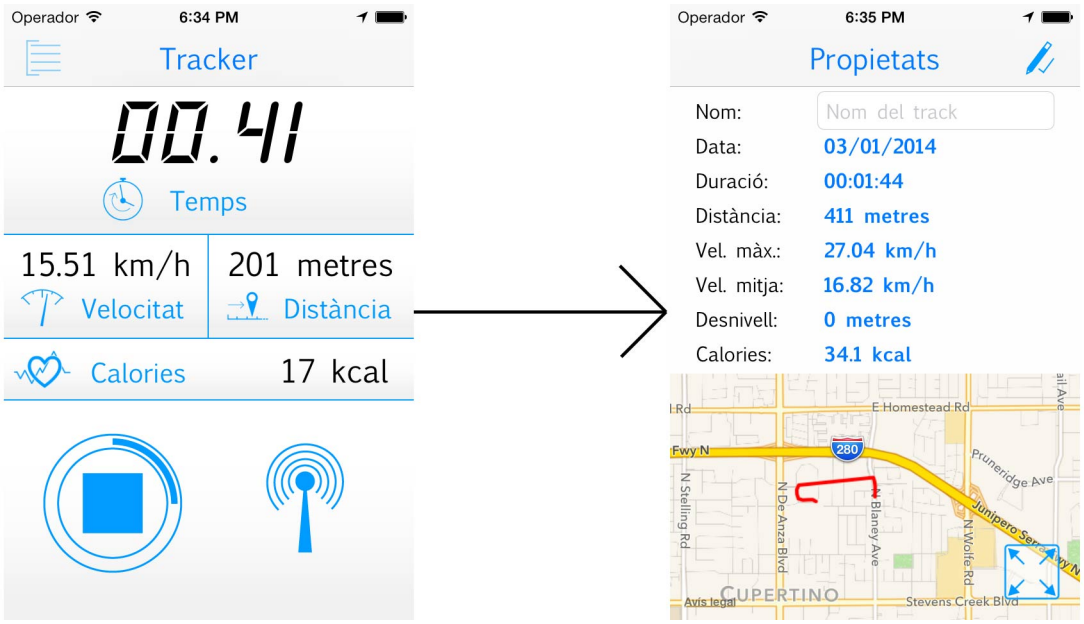


**Figura 24:** Captura de pantalla del tracker

El següent apartat del menú mostra una llisa de les rutes efectuades en forma de llista. És una llista editable, així que si se'n vol eliminar una només cal clicar el botó dret de la barra de navegació i eliminar-la. Un cop seleccionada una ruta s'accedeix a la respectiva informació, igual que un cop acabada la grabació.

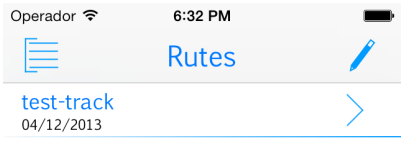


**Figura 25:** Captura de pantalla de rutes

Com es pot apreciar a la següent captura de pantalla, es mostren les estadístiques globals de l'usuari. El nombre de rutes, el la durada total, la distància recorreguda, la velocitat màxima i mitjana, la diferència d'alçada i per últim les calories cremades. Ja sigui en pantalla d'*iPhone* 5 o d'*iPhone* 4 o posterior, s'han hagut de visualitzar els elements en forma de *scroll*.

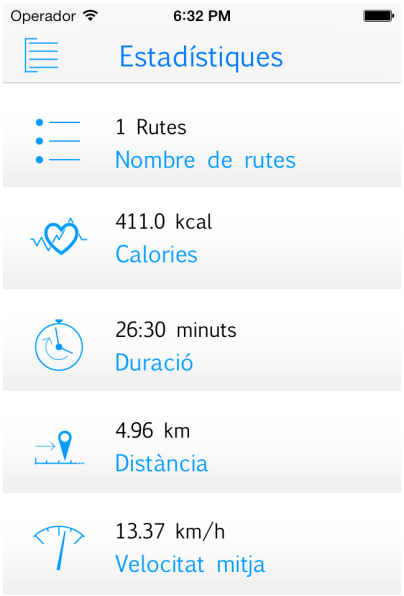


**Figura 26:** Captura de pantalla d'estadístiques

Per últim, informació sobre l'autor d'aquesta memòria i únic desenvolupador de l'aplicació. El nom, l'adreça de contacte i un enllaç web, que clicant permet accedir directament a la pàgina en qüestió. Per el moment només conté la imatge corporativa de l'aplicació, però en un futur s'implementaran les funcinoalitats web.

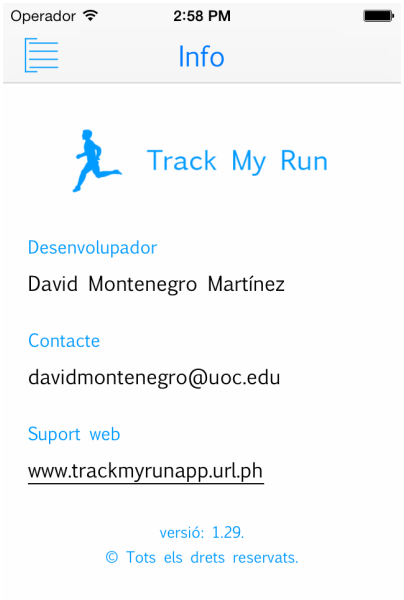


**Figura 27:** Captura de pantalla d'informació

# 12 Anàlisi econòmic

Aquesta secció descriu el pressupost que ha suposat dur a terme les diferents fases del projecte, ja sigui el desenvolupament, generació de recursos, disposició als repositoris, hosting i màrqueting. La següent figura desglossa els costos en funció dels diferents tipus.

| Concepte | Dedicació (hores) | Cost | Total |
|---|---|---|---|
| Definició del projecte | 45 | - | 1,800 € |
| Requisits | 20 | 40 €/h | 800 € |
| Metodologia | 10 | 40 €/h | 400 € |
| Planificació | 15 | 40 €/h | 600 € |
| Desenvolupament del projecte | 120 | - | 4,500 € |
| Introducció a la plataforma | 10 | 40 €/h | 400 € |
| Implementació | 80 | 40 €/h | 3,200 € |
| Maquetació de recursos | 20 | 30 €/h | 600 € |
| Recursos secundàris | 10 | 30 €/h | 300 € |
| Conceptes addicionals | 45 | - | 1,893 € |
| Programa de desenvolupament iOS | - | 72 € | 72 € |
| Traduccions | - | 25 € | 25 € |
| Documentació | 25 | 20 €/h | 500 € |
| Màrqueting | 20 | 30 €/h | 600 € |
| Hosting suport web | - | 8 €/mes | 96 € |
| Tasques transversals | 20 | 30 €/h | 600 € |
| | | | **8,193 €** |

**Figura 28:** Pressupost del projecte

El pressupost de la figura adjunta descriu el cost de totes les fases del projecte sense tenir en compte el temps total transcorregut, sinó que parteix de les hores dedicades a cada concepte/tasca. Per tal de situar cadascuna de les tasques en una línia temporal, es pot superposar amb la planificació (tot i no ser 100% paralel·les).

Es pot veure que al pressupost adjunt hi ha tasques transversals a les que no se li assigna una dedicació, doncs són tasques que només suposen un cost específic per dur a terme la tasca, ja sigui mensual o anualment. En el cas del programa de desenvolupament a iOS, el cost és anual, les traduccions (tal i com s'han definit prèviament en anteriors seccions) és el cost unitàri, i el *hosting* és en base un any. Tots els altres aspectes parteixen de la fòrmula dedicació per cost, i un sumatori de les xifres resultants obtenint el cost total del projecte.

Des d'una perspectiva a llarg termini, caldria incloure el manteniment/millores de l'aplicació (en major part causat pel futur *feedback* dels usuaris), i un pla de màrqueting desglossat per objectius.

# 13 Conclusions i treball futur

## 13.1 Conclusions

Com s'ha definit a la secció 3, el projecte es regia per uns objectius, i les conclusions han de basar-se en l'acompliment d'aquests. Per una banda, les vibracions són positives i es pot considerar que s'han complert tots els objectius tant principals com secundaris.

Per una banda, sobre el context actual del mercat, s'ha obtingut una aplicació bàsica però molt funcional, seguint les tendències de les aplicacions pioneres del seu estil, aconseguint una navegabilitat innovadora i una total adaptació als nous dispositius.

En l'última entrega s'han afegit algunes millores a l'aplicació inicial, i la seva adaptació ha estat espectacularment senzilla, doncs l'aplicació segueix una estructura modular altament adaptable a noves necessitats.

El procés d'implantació és l'únic que no s'ha dut a terme, a part que *Apple* ha restringit l'accés al repositori d'aplicacions durant el nadal (no es poden donar d'alta noves apps, ni donar de baixa ni modificar-ne el seu estat), el procés de revisió per part d'*Apple* dura aproximadament una setmana (veure l'annex), i s'hauria d'esperar un temps significatiu per veure l'impacte de l'aplicació al mercat i actuar segons el *feedback* dels usuaris, a més de desenvolupar mòduls addicionals per aportar valor afegit a l'aplicació.

Pel que fa els objectius secundaris (no per això menys importants), des d'un punt de vista més tècnic, s'han complert els objectius tals com la estabilitat, la precisió i l'estètica.

Un cop acabada la fase de *testing*, s'ha pugut confirmar que l'aplicació no conté errors (o almenys no s'han trobat en tota la fase), i funciona de manera òptima amb tots els dispositius. D'altra banda, un dels seus punts forts és la precisió de les dades que genera, factor el qual les dades de l'usuari ajuden, igual que el filtratge de la localització rebuda. I per últim, disposa d'una estètica molt acurada i uniforme, igual que una usabilitat i navegabilitat tant innovadores com intuïbles. Personalment, crec que un dels factors que ha ajudat a quadrar aquest aspecte, és el fet d'haver sigut tant desenvolupador com dissenyador, doncs sabia perfectament el necessitava i de quin procés s'havia de seguir per adaptar-ho amb la màxima precisió.

Pel que fa la documentació de la memòria, ha sigut un altre aspecte que he considerat molt important. Ha estat generada mitjançant LaTeX, un llenguatge amb el que tenia moltes ganes d'interactuar. El resultat, crec que impecable, només cal revisar el format de la documentació de dalt a baix i no percebre'n cap error.

Al següent apartat es detallen els passos a seguir del projecte.

## 13.2 Treball futur

Aquest apartat detalla quines podrien ser les millores del projecte un cop acabat i lliurada la seva primera versió. Un cop acabat aquest procés, cal implantar el producte i fer-lo econòmicament viable.

Existeixen vàries alternatives a l'hora de rendibilitzar aplicacions:

- **Publicitat**: És una de les vies més comunes. S'allibera una aplicació totalment gratuïta que ofereix totes les seves funcionalitats, buscant una diferenciació de competitivitat davant les aplicacions similars, i un cop assoleix un cert nombre d'usuaris, es busquen injeccions de capital mitjançant la publicitat de tercers.
- **In-App Purchases**: També anomenat mòduls de pagament. Com bé diu el seu nom, se sol oferir una aplicació bàsica i funcional, però aquesta no disposa de la seva totalitat de funcions. Així que un cop l'usuari es familiaritza amb l'aplicació, per tal d'obtenir totes les funcionalitats i poder exprimir-la al 100%, ha de dur a terme pagaments que habilitaran l'accés a les funcionalitats extra. També és una de les vies més comunes.
- **Pagament**: Les aplicacions de pagament són cada vegada més escasses, i encara més pels usuaris que utilitzen *smartphones* amb sistema operatiu Android. Acostumen a ser jocs 3D, paquets d'ofimàtica o similars.

El plantejament de cares a les següents fases, és implementar possibles millores a l'aplicació ja existent: Possibilitat d'identificació de diferents usuaris, plataforma de suport web funcional, millora en termes d'usabilitat, interfícies més explícites, etc.

Un cop assolides aquestes fites, el següent pas seria desenvolupar els moduls de pagament, una de les vies anteriorment explicades, i possiblement, la més ben aprofitada en el nostres cas.

Possibles mòduls de pagament (o *In-App Purchases*):

- **Mapes off-line**: Habilitar l'accés a mapes sense connexió a Internet. Al descarregar el mòdul, caldria descarregar també tots i cada un dels *tiles* (fragments dels mapes), i guardar-los en local, tenint-los sempre accessibles.
- **Sonorització**: Mentre es graba una ruta, habilitar l'accés a la llibreria musical del dispositiu, poder generar llistes de reproducció, etc. També es podria estudiar l'opció d'afegir un suport sonor a les propietats en temps real del track, ja que possiblement quan es duu a terme l'enregistrament dels punts l'usuari no estigui visualitzant la pantalla. Es podrien recordar dades tals com la velocitat mitjana, la distància recorreguda, les calories cremades, ... de manera que el corredor sap a quin ritme va, i li permet regular la intensitat que ha de seguir per arribar al seu objectiu. Podria activar-se per intervals de temps, per distància o fins i tot per diferència en el desnivell.

- **Entrenador personal**: Incloure un nou apartat al menú que permeti generar un calendari amb la intensitat/duració dels entrenaments per arribar als objectius (prèviament definits per l'usuari). Variar el calendari en funció dels entrenos efectuats tot i no seguir les pautes (previsió i planificació dinàmica). Habilitar també la visualització de la intensitat dels entrenaments a través de gràfiques.

# 14 Codi font

En aquesta secció s'imprimeixen els arxius creats pel desenvolupament de l'aplicació. Només s'inclouen els fitxers d'implementació: els fitxers de declaració contenen només la declaració dels mètodes i dels atributs, doncs el fitxer d'implementació ja els reflexa.

**InitViewController.m**

```
1  //
2  //  InitViewController.m
3  //  TrackMyRun
4  //
5  //  Created by David on 10/9/13.
6  //  Copyright (c) 2013 David. All rights reserved.
7  //
8
9  #import "InitViewController.h"
10 #import "SQLiteManager.h"
11
12 @interface InitViewController ()
13
14 @end
15
16 @implementation InitViewController
17
18 - (id)initWithNibName:(NSString *)nibNameOrNil bundle:(NSBundle *)
       nibBundleOrNil
19 {
20     self = [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
21     if (self) {
22         // Custom initialization
23     }
24     return self;
25 }
26
27 - (void)viewDidLoad
28 {
29     [super viewDidLoad];
30    // Do any additional setup after loading the view.
31     NSString *user_query = @"SELECT * FROM user";
32     NSArray *user_data = [[SQLiteManager singleton]executeSql:user_query];
33     if ([user_data count] > 0) {
34         if ([[[user_data objectAtIndex:0]objectForKey:@"name"]isEqualToString
       :@""]) {
35             [self setTopViewController:[self.storyboard
       instantiateViewControllerWithIdentifier:@"ProfileNavigation"]];
36         }
37         else {
38             [self setTopViewController:[self.storyboard
```

```
    instantiateViewControllerWithIdentifier:@"TrackerNavigation"]];
39          }
40      }
41      else {
42          [self setTopViewController:[self.storyboard
    instantiateViewControllerWithIdentifier:@"ProfileNavigation"]];
43      }
44  }
45
46  - (void)didReceiveMemoryWarning
47  {
48      [super didReceiveMemoryWarning];
49      // Dispose of any resources that can be recreated.
50  }
51
52  @end
```

**MenuViewController.m**

```
1   //
2   //  MenuViewController.m
3   //  TrackMyRun
4   //
5   //  Created by David on 10/9/13.
6   //  Copyright (c) 2013 David. All rights reserved.
7   //
8
9   #import "MenuViewController.h"
10  #import "ECSlidingViewController.h"
11
12
13  @interface MenuViewController ()
14
15  @property (strong, nonatomic) NSArray *menu_identifiers;
16  @property (strong, nonatomic) NSArray *menu_titles;
17  @property (strong, nonatomic) NSArray *menu_images;
18
19  @end
20
21  @implementation MenuViewController
22
23  @synthesize menu_identifiers;
24  @synthesize menu_titles;
25  @synthesize menu_images;
26
27  - (id)initWithStyle:(UITableViewStyle)style
28  {
29      self = [super initWithStyle:style];
30      if (self) {
31          menu_identifiers = [[NSArray alloc]init];
32          menu_titles = [[NSArray alloc]init];
```

```
33          menu_images = [[NSArray alloc]init];
34      }
35      return self;
36  }
37
38  - (void)viewDidLoad
39  {
40      [super viewDidLoad];
41
42
43      // Uncomment the following line to preserve selection between
        presentations.
44      // self.clearsSelectionOnViewWillAppear = NO;
45
46      // Uncomment the following line to display an Edit button in the
        navigation bar for this view controller.
47      // self.navigationItem.rightBarButtonItem = self.editButtonItem;
48      NSArray *identifiers = [[NSArray alloc]initWithObjects:@"Logo", @"Tracker
        ", @"Routes", @"Statistics", @"Profile", @"About", nil];
49      [self setMenu_identifiers:identifiers];
50
51      NSArray *titles = [[NSArray alloc]initWithObjects:@"Logo",
        NSLocalizedString(@"TRACKER", nil), NSLocalizedString(@"ROUTES", nil),
        NSLocalizedString(@"STATISTICS", nil), NSLocalizedString(@"PROFILE", nil)
        , NSLocalizedString(@"ABOUT", nil), nil];
52      [self setMenu_titles:titles];
53
54      NSArray *images = [[NSArray alloc]initWithObjects:@"Logo", @"
        icon_menu_tracker.png", @"icon_menu_routes.png", @"icon_menu_statistics.
        png",  @"icon_menu_user.png", @"icon_menu_about.png", nil];
55      [self setMenu_images:images];
56
57      [self.slidingViewController setAnchorRightRevealAmount:200.0f];
58      [self.slidingViewController setUnderLeftWidthLayout:ECFullWidth];
59  }
60
61  - (void)didReceiveMemoryWarning
62  {
63      [super didReceiveMemoryWarning];
64      // Dispose of any resources that can be recreated.
65  }
66
67  -(void)viewWillAppear:(BOOL)animated {
68      [super viewWillAppear:YES];
69      self.tableView.frame = CGRectMake(0, 0, self.tableView.frame.size.width,
        self.tableView.frame.size.height);
70  }
71
72  #pragma mark - Table view data source
73
74  - (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
```

```
75 {
76 //#warning Potentially incomplete method implementation.
77     // Return the number of sections.
78     return 1;
79 }
80 
81 -(CGFloat) tableView:(UITableView *)tableView heightForRowAtIndexPath:(
       NSIndexPath *)indexPath {
82     if (indexPath.row != 0) {
83         return 44.0;
84     }
85     else {
86         return 64.0;
87     }
88 }
89 
90 - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(
       NSInteger)section
91 {
92 //#warning Incomplete method implementation.
93     // Return the number of rows in the section.
94     return [self.menu_titles count];
95 }
96 
97 - (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath
       :(NSIndexPath *)indexPath
98 {
99     NSInteger row = indexPath.row;
100    NSString *cellIdentifier = @"Cell";
101    UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:
       cellIdentifier];
102    if (cell == nil) {
103        cell = [[UITableViewCell alloc]initWithStyle:
       UITableViewCellStyleSubtitle reuseIdentifier:cellIdentifier];
104    }
105 
106    if (row == 0) {
107        // Configure first cell.
108        // height: 64
109        // Cell background view
110        UIImageView *background = [[UIImageView alloc]initWithImage:[UIImage
       imageNamed:@"icon_menu_app.png"]];
111        [cell setBackgroundView:background];
112        [cell setBackgroundColor:[UIColor darkGrayColor]];
113 
114        // Cell image view properties
115        [cell.imageView setImage:[UIImage imageNamed:[self.menu_images
       objectAtIndex:row]]];
116        [cell.imageView setContentMode:UIViewContentModeScaleAspectFill];
117        [cell setSelectionStyle:UITableViewCellSelectionStyleNone];
118    }
```

```
119     else {
120         // Cell background view
121         UIImageView *background = [[UIImageView alloc]initWithImage:[UIImage
    imageNamed:@"resource_menu_background.png"]];
122         [cell setBackgroundView:background];
123         [cell setBackgroundColor:[UIColor darkGrayColor]];
124 
125         // Cell image view properties
126         [cell.imageView setImage:[UIImage imageNamed:[self.menu_images
    objectAtIndex:row]]];
127         [cell.imageView setContentMode:UIViewContentModeScaleAspectFill];
128 
129         // Cell text label properties
130         [cell.textLabel setText:[NSString stringWithFormat:@"%@", [self.
    menu_titles objectAtIndex:row]]];
131         [cell.textLabel setTextColor:[UIColor whiteColor]];
132         [cell.textLabel setFont:[UIFont fontWithName:@"Euphemia UCAS" size
    :16]];
133 
134     }
135 
136     return cell;
137 }
138 
139 - (void)tableView:(UITableView *)tableView didSelectRowAtIndexPath:(
    NSIndexPath *)indexPath {
140     if (indexPath.row != 0) {
141         NSString *identifier = [NSString stringWithString:[self.
    menu_identifiers objectAtIndex:indexPath.row]];
142         identifier = [identifier stringByAppendingString:@"Navigation"];
143         UIViewController *newTopViewController = [self.storyboard
    instantiateViewControllerWithIdentifier:identifier];
144 
145         [self.slidingViewController anchorTopViewOffScreenTo:ECRight
    animations:nil onComplete:^{
146             CGRect frame = self.slidingViewController.topViewController.view.
    frame;
147             [self.slidingViewController setTopViewController:
    newTopViewController];
148             [self.slidingViewController.topViewController.view setFrame:frame
    ];
149             [self.slidingViewController resetTopView];
150         }];
151     }
152 
153 }
154 
155 /*
156 // Override to support conditional editing of the table view.
157 - (BOOL)tableView:(UITableView *)tableView canEditRowAtIndexPath:(NSIndexPath
     *)indexPath
```

```
158 {
159     // Return NO if you do not want the specified item to be editable.
160     return YES;
161 }
162 */
163 
164 /*
165 // Override to support editing the table view.
166 - (void)tableView:(UITableView *)tableView commitEditingStyle:(
    UITableViewCellEditingStyle)editingStyle forRowAtIndexPath:(NSIndexPath
    *)indexPath
167 {
168     if (editingStyle == UITableViewCellEditingStyleDelete) {
169         // Delete the row from the data source
170         [tableView deleteRowsAtIndexPaths:@[indexPath] withRowAnimation:
    UITableViewRowAnimationFade];
171     }
172     else if (editingStyle == UITableViewCellEditingStyleInsert) {
173         // Create a new instance of the appropriate class, insert it into the
     array, and add a new row to the table view
174     }
175 }
176 */
177 
178 /*
179 // Override to support rearranging the table view.
180 - (void)tableView:(UITableView *)tableView moveRowAtIndexPath:(NSIndexPath *)
    fromIndexPath toIndexPath:(NSIndexPath *)toIndexPath
181 {
182 }
183 */
184 
185 /*
186 // Override to support conditional rearranging of the table view.
187 - (BOOL)tableView:(UITableView *)tableView canMoveRowAtIndexPath:(NSIndexPath
     *)indexPath
188 {
189     // Return NO if you do not want the item to be re-orderable.
190     return YES;
191 }
192 */
193 
194 /*
195 #pragma mark - Navigation
196 
197 // In a story board-based application, you will often want to do a little
    preparation before navigation
198 - (void)prepareForSegue:(UIStoryboardSegue *)segue sender:(id)sender
199 {
200     // Get the new view controller using [segue destinationViewController].
201     // Pass the selected object to the new view controller.
```

```
}

 */

@end
```

**ProfileViewController.m**

```
//
//  ProfileViewController.m
//  TrackMyRun
//
//  Created by David on 10/23/13.
//  Copyright (c) 2013 David. All rights reserved.
//
#define kOFFSET_FOR_KEYBOARD        216.0f

#import "ProfileViewController.h"
#import "ECSlidingViewController.h"
#import "MenuViewController.h"
#import "NavigationToolbarBuilder.h"
#import "SQLiteManager.h"


@interface ProfileViewController ()

@end

@implementation ProfileViewController

- (void)viewDidLoad
{
    [super viewDidLoad];
   // Do any additional setup after loading the view.

    // Init data.
    editing = NO;
    is_up = NO;

    // View properties.
    [self.view.layer setShadowOpacity:0.75f];
    [self.view.layer setShadowRadius:10.0f];
    [self.view.layer setShadowColor:[UIColor blackColor].CGColor];

    if (![self.slidingViewController.underLeftViewController isKindOfClass:[
    MenuViewController class]]) {
        [self.slidingViewController setUnderLeftViewController:[self.
    storyboard instantiateViewControllerWithIdentifier:@"Menu"]];
    }

    [self.view addGestureRecognizer:self.slidingViewController.panGesture];
```

```
42
43      [self loadNavigationElements];
44      [self loadInterfaceElements];
45
46      NSArray *user_data = [self getUserData];
47      if ([user_data count] > 0) {
48          if ([[[user_data objectAtIndex:0]objectForKey:@"name"]isEqualToString
        :@""]) {
49              [self editUserData];
50          }
51          else {
52              [self showUserData];
53          }
54      }
55      else {
56          [self editUserData];
57      }
58  }
59
60  -(IBAction)openMenu:(id)sender {
61      [self.slidingViewController anchorTopViewTo:ECRight];
62  }
63
64  -(IBAction)changeEditingMode:(id)sender {
65      if (editing) {
66          [self updateUserDataToDatabase];
67          [self showUserData];
68      }
69      else {
70          [self editUserData];
71      }
72  }
73
74  -(NSArray*)getUserData {
75      NSString *user_query = @"SELECT * FROM user";
76      return [[SQLiteManager singleton]executeSql:user_query];
77  }
78
79  -(void)updateUserDataToDatabase {
80      NSString *sex = [NSString stringWithFormat:@"Female"];
81      if(segmented_sex.selectedSegmentIndex == 0) {
82          sex = @"Male";
83      }
84      NSMutableDictionary *user_data = [NSMutableDictionary
        dictionaryWithObjectsAndKeys:@"1", @"id", [textfield_username text], @"
        name", sex, @"sex", [textfield_age text], @"age", [textfield_weight text
        ], @"weight", nil];
85      [[SQLiteManager singleton]save:user_data into:@"user"];
86  }
87
88  -(void)loadNavigationElements {
```

```
89      // Build navigation toolbar items.
90      [self.navigationItem setLeftBarButtonItem:[NavigationToolbarBuilder
        createMenuBarButtonAtTarget:self]];
91      [self.navigationItem setTitleView:[NavigationToolbarBuilder
        createTitleBarButtonWithName:NSLocalizedString(@"PROFILE", nil)]];
92
93      UIButton *edit_button = [UIButton buttonWithType:UIButtonTypeCustom];
94      [edit_button setFrame:CGRectMake(0, 0, 30, 30)];
95      [edit_button setImage:[UIImage imageNamed:@"icon_edit.png"] forState:
        UIControlStateNormal];
96      [edit_button addTarget:self action:@selector(changeEditingMode:)
        forControlEvents:UIControlEventTouchUpInside];
97      UIBarButtonItem *edit_bar_button = [[UIBarButtonItem alloc]
        initWithCustomView:edit_button];
98      [self.navigationItem setRightBarButtonItem:edit_bar_button];
99  }
100
101 -(void)loadInterfaceElements {
102     // Interface elements.
103     [label_username setText:NSLocalizedString(@"USERNAME", nil)];
104     [label_sex setText:NSLocalizedString(@"SEX", nil)];
105     [label_age setText:NSLocalizedString(@"AGE", nil)];
106     [label_weight setText:NSLocalizedString(@"WEIGHT", nil)];
107
108     NSDictionary *attributes = [NSDictionary dictionaryWithObjectsAndKeys:[
        UIFont fontWithName:@"Euphemia UCAS" size:16], UITextAttributeFont, [
        UIColor colorWithRed:0.0/255.0 green:122.0/255.0 blue:255.0/255.0 alpha
        :1.0], UITextAttributeTextColor, nil];
109     NSDictionary *attributes_2 = [NSDictionary dictionaryWithObjectsAndKeys:[
        UIFont fontWithName:@"Euphemia UCAS" size:16], UITextAttributeFont, [
        UIColor whiteColor], UITextAttributeTextColor, nil];
110     [segmented_sex setTitleTextAttributes:attributes forState:
        UIControlStateNormal];
111     [segmented_sex setTitleTextAttributes:attributes_2 forState:
        UIControlStateHighlighted];
112     [segmented_sex setTitle:NSLocalizedString(@"MAN", nil) forSegmentAtIndex
        :0];
113     [segmented_sex setTitle:NSLocalizedString(@"WOMAN", nil)
        forSegmentAtIndex:1];
114
115     [textfield_username setFont:[UIFont fontWithName:@"Euphemia UCAS" size
        :16]];
116     [textfield_username setTextColor:[UIColor colorWithRed:0.0/255.0 green
        :122.0/255.0 blue:255.0/255.0 alpha:1.0]];
117     [textfield_age setFont:[UIFont fontWithName:@"Euphemia UCAS" size:16]];
118     [textfield_age setTextColor:[UIColor colorWithRed:0.0/255.0 green
        :122.0/255.0 blue:255.0/255.0 alpha:1.0]];
119     [textfield_weight setFont:[UIFont fontWithName:@"Euphemia UCAS" size
        :16]];
120     [textfield_weight setTextColor:[UIColor colorWithRed:0.0/255.0 green
        :122.0/255.0 blue:255.0/255.0 alpha:1.0]];
```

```
121
122     [label_data_usage setText:NSLocalizedString(@"DATA_ALERT", nil)];
123 }
124
125 -(void)showUserData {
126     editing = NO;
127     NSArray *user_data = [self getUserData];
128     if ([user_data count] > 0) {
129         NSDictionary *user = [user_data objectAtIndex:0];
130         if ((NSNull*)[user objectForKey:@"name"] != [NSNull null] && ![[user
    objectForKey:@"name"]isEqualToString:@""]){
131             [label_username_value setText:[user objectForKey:@"name"]];
132         }
133         else  {
134             [label_username_value setText:@"-"];
135         }
136         if ((NSNull*)[user objectForKey:@"sex"] != [NSNull null] && ![[user
    objectForKey:@"sex"]isEqualToString:@""]) {
137             if ([[user objectForKey:@"sex"]isEqualToString:@"Female"]) {
138                 [segmented_sex setSelectedSegmentIndex:1];
139             }
140             else {
141                 [segmented_sex setSelectedSegmentIndex:0];
142             }
143         }
144         if ((NSNull*)[user objectForKey:@"age"] != [NSNull null] && ![[user
    objectForKey:@"age"]isEqualToString:@""]) {
145             NSString *age_value = [[NSString alloc]initWithFormat:@"%@ %@", [
    user objectForKey:@"age"], NSLocalizedString(@"YEARS", nil)];
146             [label_age_value setText:age_value];
147         }
148         else {
149             [label_age_value setText:@"-"];
150         }
151         if ((NSNull*)[user objectForKey:@"weight"] != [NSNull null] && ![[
    user objectForKey:@"weight"]isEqualToString:@""]) {
152             NSString *weight_value = [[NSString alloc]initWithFormat:@"%@ kg.
    ", [user objectForKey:@"weight"]];
153             [label_weight_value setText:weight_value];
154         }
155         else  {
156             [label_weight_value setText:@"-"];
157         }
158     }
159
160     [textfield_username setHidden:YES];
161     [label_username_value setHidden:NO];
162     [textfield_age setHidden:YES];
163     [label_age_value setHidden:NO];
164     [textfield_weight setHidden:YES];
165     [label_weight_value setHidden:NO];
```

```
166
167     UIButton *edit_button = [UIButton buttonWithType:UIButtonTypeCustom];
168     [edit_button setFrame:CGRectMake(0, 0, 30, 30)];
169     [edit_button setImage:[UIImage imageNamed:@"icon_edit.png"] forState:
        UIControlStateNormal];
170     [edit_button addTarget:self action:@selector(changeEditingMode:)
        forControlEvents:UIControlEventTouchUpInside];
171     UIBarButtonItem *edit_bar_button = [[UIBarButtonItem alloc]
        initWithCustomView:edit_button];
172     [self.navigationItem setRightBarButtonItem:edit_bar_button];
173 }
174
175 -(void)editUserData {
176     editing = YES;
177     NSArray *user_data = [self getUserData];
178     if ([user_data count] > 0) {
179         NSDictionary *user = [user_data objectAtIndex:0];
180         if ((NSNull*)[user objectForKey:@"name"] != [NSNull null] && ![[user
        objectForKey:@"name"]isEqualToString:@""]) {
181             [textfield_username setText:[user objectForKey:@"name"]];
182         }
183         else  {
184             [textfield_username setPlaceholder:NSLocalizedString(@"ENTER_NAME
        ", nil)];
185         }
186         if ((NSNull*)[user objectForKey:@"sex"] != [NSNull null]) {
187             if ([[user objectForKey:@"sex"]isEqualToString:@"Female"]) {
188                 [segmented_sex setSelectedSegmentIndex:1];
189             }
190             else {
191                 [segmented_sex setSelectedSegmentIndex:0];
192             }
193         }
194         if ((NSNull*)[user objectForKey:@"age"] != [NSNull null] && ![[user
        objectForKey:@"age"]isEqualToString:@""]) {
195             [textfield_age setText:[user objectForKey:@"age"]];
196         }
197         else  {
198             [textfield_age setPlaceholder:NSLocalizedString(@"ENTER_AGE", nil
        )];
199         }
200         if ((NSNull*)[user objectForKey:@"weight"] != [NSNull null] && ![[
        user objectForKey:@"weight"]isEqualToString:@""]) {
201             [textfield_weight setText:[user objectForKey:@"weight"]];
202         }
203         else  {
204             [textfield_weight setPlaceholder:NSLocalizedString(@"ENTER_WEIGHT
        ", nil)];
205         }
206     }
207     else {
```

```
208         [textfield_username setPlaceholder:NSLocalizedString(@"ENTER_NAME",
    nil)];
209         [textfield_age setPlaceholder:NSLocalizedString(@"ENTER_AGE", nil)];
210         [textfield_weight setPlaceholder:NSLocalizedString(@"ENTER_WEIGHT",
    nil)];
211     }
212
213     [label_username_value setHidden:YES];
214     [textfield_username setHidden:NO];
215     [label_age_value setHidden:YES];
216     [textfield_age setHidden:NO];
217     [label_weight_value setHidden:YES];
218     [textfield_weight setHidden:NO];
219
220     UIButton *edit_button = [UIButton buttonWithType:UIButtonTypeCustom];
221     [edit_button setFrame:CGRectMake(0, 0, 30, 30)];
222     [edit_button setImage:[UIImage imageNamed:@"icon_edit_done.png"] forState
    :UIControlStateNormal];
223     [edit_button addTarget:self action:@selector(changeEditingMode:)
    forControlEvents:UIControlEventTouchUpInside];
224     UIBarButtonItem *edit_bar_button = [[UIBarButtonItem alloc]
    initWithCustomView:edit_button];
225     [self.navigationItem setRightBarButtonItem:edit_bar_button];
226 }
227
228 - (void)didReceiveMemoryWarning
229 {
230     [super didReceiveMemoryWarning];
231     // Dispose of any resources that can be recreated.
232 }
233
234 #pragma  UITextField delegate methods
235 -(BOOL)textFieldShouldReturn:(UITextField*)textField{
236     [textField resignFirstResponder];
237     return YES;
238 }
239
240 - (void)touchesBegan:(NSSet*)touches withEvent:(UIEvent*)event {
241     [textfield_username resignFirstResponder];
242     [textfield_age resignFirstResponder];
243     [textfield_weight resignFirstResponder];
244 }
245
246 - (void)textFieldDidEndEditing:(UITextField *)textField {
247     if (is_up) {
248         CGRect rect;
249         // If the ios version is less than 7.0.
250         if ((([[[UIDevice currentDevice] systemVersion] compare:@"7.0" options
    :NSNumericSearch] == NSOrderedAscending)) {
251             rect = CGRectMake(0, 0, self.view.frame.size.width, self.view.
    frame.size.height);
```

```
252         }
253         else {
254             rect = CGRectMake(0, self.navigationController.toolbar.bounds.
    size.height + [UIApplication sharedApplication].statusBarFrame.size.
    height, self.view.frame.size.width, self.view.frame.size.height);
255         }
256         [UIView beginAnimations:nil context:NULL];
257         [UIView setAnimationDuration:0.3];
258         [self.view setFrame:rect];
259         [UIView commitAnimations];
260         is_up = NO;
261     }
262 }
263 
264 -(BOOL)textFieldShouldBeginEditing:(UITextField*)textField {
265     if (textField.frame.origin.y + textField.frame.size.height > self.view.
    bounds.size.height - 216) {
266         is_up = YES;
267         double offset = self.view.bounds.size.height - 216 - textField.frame.
    origin.y - textField.frame.size.height - 20;
268         CGRect rect = CGRectMake(0, offset, self.view.frame.size.width, self.
    view.frame.size.height);
269         [UIView beginAnimations:nil context:NULL];
270         [UIView setAnimationDuration:0.3];
271         [self.view setFrame:rect];
272         [UIView commitAnimations];
273     }
274     return YES;
275 }
276 
277 - (BOOL)textField:(UITextField *)textField shouldChangeCharactersInRange:(
    NSRange)range replacementString:(NSString *)string {
278     if ([textField tag] == 0) {
279         if ([textField.text length] > 29) {
280             textField.text = [textField.text substringToIndex:29-1];
281             return NO;
282         }
283     }
284     else if ([textField tag] == 1) {
285         if ([textField.text length] > 1) {
286             textField.text = [textField.text substringToIndex:2-1];
287             return NO;
288         }
289     }
290     else if ([textField tag] == 2) {
291         if ([textField.text length] > 2) {
292             textField.text = [textField.text substringToIndex:3-1];
293             return NO;
294         }
295     }
296     return YES;
```

```
297 }
298
299 @end
```

**TrackerViewController.m**

```
1  //
2  //  MainViewController.m
3  //  TrackMyRun
4  //
5  //  Created by David on 10/9/13.
6  //  Copyright (c) 2013 David. All rights reserved.
7  //
8
9  #import "TrackerViewController.h"
10 #import "ECSlidingViewController.h"
11 #import "MenuViewController.h"
12 #import "NavigationToolbarBuilder.h"
13 #import "TrackDetailViewController.h"
14
15 @interface TrackerViewController ()
16
17 @end
18
19 @implementation TrackerViewController
20
21 - (void)viewDidLoad
22 {
23     [super viewDidLoad];
24    // Do any additional setup after loading the view.
25
26     // Init data values.
27     [self initData];
28
29     // View properties.
30     [self.view.layer setShadowOpacity:0.75f];
31     [self.view.layer setShadowRadius:10.0f];
32     [self.view.layer setShadowColor:[UIColor blackColor].CGColor];
33
34     if (![self.slidingViewController.underLeftViewController isKindOfClass:[
       MenuViewController class]]) {
35         [self.slidingViewController setUnderLeftViewController:[self.
       storyboard instantiateViewControllerWithIdentifier:@"Menu"]];
36     }
37
38     [self.view addGestureRecognizer:self.slidingViewController.panGesture];
39
40     // Build navigation toolbar items.
41     [self.navigationItem setLeftBarButtonItem:[NavigationToolbarBuilder
       createMenuBarButtonAtTarget:self]];
```

```
42      [self.navigationItem setTitleView:[NavigationToolbarBuilder
        createTitleBarButtonWithName:NSLocalizedString(@"TRACKER", nil)]];
43
44      [self loadInterfaceElements];
45
46      // Refresh speed and distance when location is recieved.
47      [[NSNotificationCenter defaultCenter] addObserver:self selector:@selector
        (refreshRealTimeProperties:) name:@"tracking" object:nil];
48  }
49
50  -(void)initData {
51      button_x = button_tracking.frame.origin.x;
52      button_y = button_tracking.frame.origin.y;
53      is_tracking = NO;
54      current_time = 0;
55      current_gps_tag = 0;
56      loading = 1;
57      location_service = [[LocationService alloc]init];
58      [location_service setDelegate:self];
59      [gps_image setAlpha:0.0f];
60  }
61
62  - (void)didReceiveMemoryWarning
63  {
64      [super didReceiveMemoryWarning];
65      // Dispose of any resources that can be recreated.
66  }
67
68  -(void)loadInterfaceElements {
69      [label_duration setFont:[UIFont fontWithName:@"Euphemia UCAS" size:20]];
70      [label_duration setText:NSLocalizedString(@"TIME", nil)];
71      [label_duration_value setFont:[UIFont fontWithName:@"DBLCDTempBlack" size
        :60]];
72      [label_duration_value setText:@"00.00"];
73
74      [label_speed setFont:[UIFont fontWithName:@"Euphemia UCAS" size:20]];
75      [label_speed setText:NSLocalizedString(@"SPEED", nil)];
76      [label_speed_value setFont:[UIFont fontWithName:@"Euphemia UCAS" size
        :24]];
77      [label_speed_value setText:@"0 km/h"];
78
79
80      [label_distance setFont:[UIFont fontWithName:@"Euphemia UCAS" size:20]];
81      [label_distance setText:NSLocalizedString(@"DISTANCE_EMPTY", nil)];
82      [label_distance_value setFont:[UIFont fontWithName:@"Euphemia UCAS" size
        :24]];
83      [label_distance_value setText:@"0 m"];
84
85      [label_calories setFont:[UIFont fontWithName:@"Euphemia UCAS" size:20]];
86      [label_calories setText:NSLocalizedString(@"CALORIES_EMPTY", nil)];
```

```
87      [label_calories_value setFont:[UIFont fontWithName:@"Euphemia UCAS" size
        :24]];
88      [label_calories_value setText:@"0 kcal"];
89  }
90
91  -(void)refreshRealTimeProperties:(NSNotification*)notification {
92      NSString *speed = [[NSString alloc]initWithFormat:@"%@ km/h", [
        notification.userInfo objectForKey:@"speed"]];
93      [label_speed_value setText:speed];
94      NSString *distance = [[NSString alloc]init];
95      float current_distance = [[notification.userInfo objectForKey:@"distance"
        ]floatValue];
96      if (current_distance >= 1000) {
97          current_distance = current_distance / 1000;
98          distance = [NSString stringWithFormat:@"%.2f km", current_distance];
99      }
100     else {
101         distance = [NSString stringWithFormat:@"%.0f %@", current_distance,
        NSLocalizedString(@"METERS", nil)];
102     }
103     [label_distance_value setText:distance];
104
105     NSString *calories = [[NSString alloc]initWithFormat:@"%@ kcal", [
        notification.userInfo objectForKey:@"calories"]];
106     [label_calories_value setText:calories];
107
108 }
109
110 -(IBAction)openMenu:(id)sender {
111     [self.slidingViewController anchorTopViewTo:ECRight];
112 }
113
114 - (IBAction)changeTrackingState:(id)sender {
115     if (!is_tracking) {
116         // Start tracking.
117         is_tracking = YES;
118         timer = [NSTimer scheduledTimerWithTimeInterval:1 target:self
        selector:@selector(showTime) userInfo:nil repeats:YES];
119         timer2 = [NSTimer scheduledTimerWithTimeInterval:0.2 target:self
        selector:@selector(changeGPSImage) userInfo:nil repeats:YES];
120         [button_tracking setImage:[UIImage imageNamed:@"button_stop.png"]
        forState:UIControlStateNormal];
121         [self.navigationItem.leftBarButtonItem setEnabled:NO];
122         [self.view removeGestureRecognizer:self.slidingViewController.
        panGesture];
123         [self moveButtonAnimated];
124         [UIView animateWithDuration:2.0f animations:^{
125             [gps_image setAlpha:1.0f];
126         }];
127         [location_service start];
128     }
```

```
129     else {
130         is_tracking = NO;
131         [timer invalidate];
132         [timer2 invalidate];
133         [location_service setDuration:current_time];
134         [loading_image setHidden:YES];
135         current_time = 0;
136         loading = 1;
137         current_gps_tag = 0;
138         [gps_image setAlpha:0.0f];
139         [button_tracking setImage:[UIImage imageNamed:@"button_start.png"]
    forState:UIControlStateNormal];
140         [button_tracking setFrame:CGRectMake(button_x, button_y,
    button_tracking.frame.size.width, button_tracking.frame.size.height)];
141         [self changeGPSImage];
142         [self.navigationItem.leftBarButtonItem setEnabled:YES];
143         [self.view addGestureRecognizer:self.slidingViewController.panGesture
    ];
144         BOOL valid_track = [location_service stop];
145 
146         // Reset tracking interface elements.
147         [label_duration_value setText:@"00.00"];
148         [label_speed_value setText:@"0 km/h"];
149         [label_distance_value setText:@"0 m"];
150 
151         if (valid_track) {
152             // Push the next view controller.
153             NSString *identifier = @"TrackDetail";
154             TrackDetailViewController *track_detail = [self.storyboard
    instantiateViewControllerWithIdentifier:identifier];
155             [track_detail setTrack_id:location_service.track.identifier];
156             [track_detail setFrom_tracking:YES];
157             [self.navigationController pushViewController:track_detail
    animated:NO];
158         }
159     }
160 }
161 
162 -(void)showTime {
163     // Set seconds, minutes and hours.
164     current_time++;
165     int seconds = current_time % 60;
166     int minutes = (current_time / 60) % 60;
167     int hours = (current_time / 3600) % 60;
168     if (hours > 0) {
169         [label_duration_value setText:[NSString stringWithFormat:@"%.2d.%.2d
    .%.2d", hours, minutes, seconds]];
170     }
171     else {
172         [label_duration_value setText:[NSString stringWithFormat:@"%.2d.%.2d"
    , minutes, seconds]];
```

```
173     }
174     // Set the loading image
175     [loading_image setHidden:NO];
176     NSString *image_name = [NSString stringWithFormat:@"loading_%i.png",
        loading];
177     [loading_image setImage:[UIImage imageNamed:image_name]];
178     if (loading < 4) {
179         loading++;
180     }
181     else {
182         loading = 1;
183     }
184 }
185 
186 -(void)changeGPSImage {
187     NSString *image_name = [NSString stringWithFormat:@"icon_gps_%i.png",
        current_gps_tag];
188     [gps_image setImage:[UIImage imageNamed:image_name]];
189     if (current_gps_tag < 4) {
190         current_gps_tag++;
191     }
192     else {
193         current_gps_tag = 0;
194     }
195 }
196 
197 -(void)moveButtonAnimated {
198     [UIView animateWithDuration:1.0f animations:^{
199         //Move the image view to 100, 100 over 10 seconds.
200         button_tracking.frame = CGRectMake(button_tracking.frame.size.width
        -65, button_tracking.frame.origin.y, button_tracking.frame.size.width,
        button_tracking.frame.size.height);
201     }];
202 }
203 
204 
205 @end
```

**TrackDetailViewController.m**

```
1  //
2  //  TrackDetailViewController.m
3  //  TrackMyRun
4  //
5  //  Created by David on 11/22/13.
6  //  Copyright (c) 2013 David. All rights reserved.
7  //
8  
9  #import "TrackDetailViewController.h"
10 #import "MenuViewController.h"
11 #import "NavigationToolbarBuilder.h"
```

```
12 #import "SQLiteManager.h"
13 #import "MapViewController.h"
14
15 @interface TrackDetailViewController ()
16
17 @end
18
19 @implementation TrackDetailViewController
20 @synthesize track_id;
21 @synthesize from_tracking;
22 @synthesize polyline;
23
24 - (void)viewDidLoad
25 {
26     [super viewDidLoad];
27    // Do any additional setup after loading the view.
28
29     // View properties.
30     [self.view.layer setShadowOpacity:0.75f];
31     [self.view.layer setShadowRadius:10.0f];
32     [self.view.layer setShadowColor:[UIColor blackColor].CGColor];
33
34     [self loadNavigationElements];
35     [self loadInterfaceElements];
36
37     editing = NO;
38     [mapview setDelegate:self];
39     [self buildPolyline];
40 }
41
42 - (void)didReceiveMemoryWarning
43 {
44     [super didReceiveMemoryWarning];
45     // Dispose of any resources that can be recreated.
46 }
47
48 -(void)loadNavigationElements {
49     // Build navigation toolbar items.
50     [self.navigationItem setTitleView:[NavigationToolbarBuilder
    createTitleBarButtonWithName:NSLocalizedString(@"PROPERTIES", nil)]];
51
52     UIButton *edit_button = [UIButton buttonWithType:UIButtonTypeCustom];
53     [edit_button setFrame:CGRectMake(0, 0, 30, 30)];
54     if (from_tracking) {
55         [self.navigationItem setHidesBackButton:YES];
56         [edit_button setImage:[UIImage imageNamed:@"icon_edit_done.png"]
    forState:UIControlStateNormal];
57         [edit_button addTarget:self action:@selector(resumeDone:)
    forControlEvents:UIControlEventTouchUpInside];
58     }
59     else {
```

```
60          [self.navigationItem setLeftBarButtonItem:[NavigationToolbarBuilder
    createBackBarButtonWithTarget:self]];
61          [edit_button setImage:[UIImage imageNamed:@"icon_edit.png"] forState:
    UIControlStateNormal];
62          [edit_button addTarget:self action:@selector(changeEditingMode:)
    forControlEvents:UIControlEventTouchUpInside];
63      }
64      UIBarButtonItem *edit_bar_button = [[UIBarButtonItem alloc]
    initWithCustomView:edit_button];
65      [self.navigationItem setRightBarButtonItem:edit_bar_button];
66  }
67
68  -(void)loadInterfaceElements {
69      NSArray *track_result = [self getTrackData];
70      NSDictionary *track_data = [track_result objectAtIndex:0];
71      // Interface elements.
72      [label_name setText:NSLocalizedString(@"NAME", nil)];
73      [label_date setText:NSLocalizedString(@"DATE", nil)];
74      [label_duration setText:NSLocalizedString(@"DURATION", nil)];
75      [label_distance setText:NSLocalizedString(@"DISTANCE", nil)];
76      [label_max_speed setText:NSLocalizedString(@"MAX_SPEED", nil)];
77      [label_mean_speed setText:NSLocalizedString(@"MEAN_SPEED", nil)];
78      [label_height_diff setText:NSLocalizedString(@"HEIGHT_DIFF", nil)];
79      [label_calories setText:NSLocalizedString(@"CALORIES", nil)];
80
81      if (from_tracking) {
82          [label_name_value setHidden:YES];
83          [textfield_name setHidden:NO];
84          [textfield_name setPlaceholder:NSLocalizedString(@"ENTER_TRACK_NAME",
     nil)];
85      }
86      else {
87          [textfield_name setHidden:YES];
88          [label_name_value setHidden:NO];
89          [label_name_value setText:[track_data objectForKey:@"name"]];
90          [self.navigationItem setTitleView:[NavigationToolbarBuilder
    createTitleBarButtonWithName:[track_data objectForKey:@"name"]]];
91      }
92      [label_date_value setText:[track_data objectForKey:@"date"]];
93      NSString *time = [self timeFormatted:[[track_data objectForKey:@"duration
    "]intValue]];
94      [label_duration_value setText:time];
95      if ([[track_data objectForKey:@"distance"]floatValue] < 1000) {
96          [label_distance_value setText:[NSString stringWithFormat:@"%@ %@", [
    track_data objectForKey:@"distance"], NSLocalizedString(@"METERS", nil)
    ]];
97      }
98      else {
99          [label_distance_value setText:[NSString stringWithFormat:@"%.2f km",
    [[track_data objectForKey:@"distance"]floatValue]/1000]];
100     }
```

```
101
102     [label_max_speed_value setText:[NSString stringWithFormat:@"%@ km/h", [
        track_data objectForKey:@"max_speed"]]];
103     [label_mean_speed_value setText:[NSString stringWithFormat:@"%@ km/h", [
        track_data objectForKey:@"mean_speed"]]];
104     [label_height_diff_value setText:[NSString stringWithFormat:@"%@ %@", [
        track_data objectForKey:@"height_diff"], NSLocalizedString(@"METERS", nil
        )]];
105     [label_calories_value setText:[NSString stringWithFormat:@"%@ kcal", [
        track_data objectForKey:@"calories"]]];
106 }
107
108 -(NSArray*)getTrackData {
109     NSString *track_query = [NSString stringWithFormat:@"SELECT * FROM track
        WHERE id = %@", track_id];
110     return [[SQLiteManager singleton]executeSql:track_query];
111 }
112
113 -(NSArray*)getLocationData {
114     NSString *points_query = [NSString stringWithFormat:@"SELECT latitude,
        longitude FROM location WHERE track = %@", track_id];
115     return [[SQLiteManager singleton]executeSql:points_query];
116 }
117
118 -(NSString*)timeFormatted:(int)totalSeconds {
119     int seconds = totalSeconds%60;
120     int minutes = (totalSeconds/60)%60;
121     int hours = totalSeconds/3600;
122
123     return [NSString stringWithFormat:@"%02d:%02d:%02d",hours, minutes,
        seconds];
124 }
125
126 -(IBAction)resumeDone:(id)sender {
127     // Save the current track name.
128     if (textfield_name.text.length == 0) {
129         [textfield_name setText:NSLocalizedString(@"UNNAMED", nil)];
130         [self.navigationItem setTitleView:[NavigationToolbarBuilder
        createTitleBarButtonWithName:NSLocalizedString(@"PROPERTIES", nil)]];
131     }
132     else {
133         [self.navigationItem setTitleView:[NavigationToolbarBuilder
        createTitleBarButtonWithName:textfield_name.text]];
134     }
135     NSMutableDictionary *track_data = [NSMutableDictionary
        dictionaryWithObjectsAndKeys:track_id, @"id", [textfield_name text], @"
        name", nil];
136     [[SQLiteManager singleton]save:track_data into:@"track"];
137     [self.navigationController popToRootViewControllerAnimated:YES];
138 }
139
```

```
140 -(IBAction)goBack:(id)sender {
141     [self.navigationController popViewControllerAnimated:YES];
142 }
143 
144 -(IBAction)changeEditingMode:(id)sender {
145     UIButton *edit_button = [UIButton buttonWithType:UIButtonTypeCustom];
146     [edit_button setFrame:CGRectMake(0, 0, 30, 30)];
147     [edit_button addTarget:self action:@selector(changeEditingMode:)
    forControlEvents:UIControlEventTouchUpInside];
148     if (editing) {
149         editing = NO;
150         [textfield_name setHidden:YES];
151         if (textfield_name.text.length == 0) {
152             [textfield_name setText:NSLocalizedString(@"UNNAMED", nil)];
153             [self.navigationItem setTitleView:[NavigationToolbarBuilder
    createTitleBarButtonWithName:NSLocalizedString(@"PROPERTIES", nil)]];
154         }
155         else {
156             [self.navigationItem setTitleView:[NavigationToolbarBuilder
    createTitleBarButtonWithName:textfield_name.text]];
157         }
158         [label_name_value setHidden:NO];
159         [label_name_value setText:textfield_name.text];
160         [edit_button setImage:[UIImage imageNamed:@"icon_edit.png"] forState:
    UIControlStateNormal];
161 
162         NSMutableDictionary *track_data = [NSMutableDictionary
    dictionaryWithObjectsAndKeys:track_id, @"id", [textfield_name text], @"
    name", nil];
163         [[SQLiteManager singleton]save:track_data into:@"track"];
164     }
165     else {
166         editing = YES;
167         [label_name_value setHidden:YES];
168         [textfield_name setHidden:NO];
169         [textfield_name setText:label_name_value.text];
170         [edit_button setImage:[UIImage imageNamed:@"icon_edit_done.png"]
    forState:UIControlStateNormal];
171 
172     }
173     UIBarButtonItem *edit_bar_button = [[UIBarButtonItem alloc]
    initWithCustomView:edit_button];
174     [self.navigationItem setRightBarButtonItem:edit_bar_button];
175 }
176 
177 - (IBAction)openFullScreenMap:(id)sender {
178     NSString *identifier = @"Map";
179     MapViewController *full_map_view = [self.storyboard
    instantiateViewControllerWithIdentifier:identifier];
180     [full_map_view setPolyline:polyline];
181     [full_map_view setRegion:region];
```

```
182     [self.navigationController pushViewController:full_map_view animated:YES
          ];
183 }
184
185 #pragma UITextFieldDelegate
186 -(BOOL)textFieldShouldReturn:(UITextField*)textField {
187     [textField resignFirstResponder];
188     return YES;
189 }
190
191 #pragma MKMapViewDelegate
192 - (MKOverlayView*)mapView:(MKMapView*)theMapView viewForOverlay:(id <
    MKOverlay>)overlay {
193     MKPolylineView* lineView = [[MKPolylineView alloc]initWithPolyline:self.
    polyline];
194     [lineView setFillColor:[UIColor whiteColor]];
195     [lineView setStrokeColor:[UIColor redColor]];
196     [lineView setLineWidth:4];
197
198     return lineView;
199 }
200
201 -(void)buildPolyline {
202     NSArray *path = [self getLocationData];
203     NSInteger number_of_steps = path.count;
204     CLLocationCoordinate2D coordinates[number_of_steps];
205
206     for (NSInteger index = 0; index < number_of_steps; index++) {
207         NSString *latitude = [[path objectAtIndex:index]objectForKey:@"
    latitude"];
208         NSString *longitude = [[path objectAtIndex:index]objectForKey:@"
    longitude"];
209         CLLocationCoordinate2D coordinate = CLLocationCoordinate2DMake([
    latitude floatValue], [longitude floatValue]);
210         coordinates[index] = coordinate;
211     }
212     polyline = [MKPolyline polylineWithCoordinates:coordinates count:
    number_of_steps];
213     MKCoordinateRegion init_region;
214     init_region.center = coordinates[0];
215     MKCoordinateSpan span;
216     // 0.001 to 120.
217     span.latitudeDelta = 0.025;
218     span.longitudeDelta = 0.025;
219     init_region.span = span;
220     region = init_region;
221     [mapview setRegion:init_region animated:YES];
222     [mapview addOverlay:polyline];
223 }
224
225 @end
```

**RoutesViewController.m**

```
1  //
2  //  RoutesViewController.m
3  //  TrackMyRun
4  //
5  //  Created by David on 10/23/13.
6  //  Copyright (c) 2013 David. All rights reserved.
7  //
8
9  #import "RoutesViewController.h"
10 #import "ECSlidingViewController.h"
11 #import "MenuViewController.h"
12 #import "NavigationToolbarBuilder.h"
13 #import "SQLiteManager.h"
14 #import "TrackDetailViewController.h"
15
16 @interface RoutesViewController ()
17
18 @end
19
20 @implementation RoutesViewController
21
22 - (void)viewDidLoad
23 {
24     [super viewDidLoad];
25    // Do any additional setup after loading the view.
26
27     // Init data values.
28     [self initData];
29
30     // View properties
31     [self.view.layer setShadowOpacity:0.75f];
32     [self.view.layer setShadowRadius:10.0f];
33     [self.view.layer setShadowColor:[UIColor blackColor].CGColor];
34
35     if (![self.slidingViewController.underLeftViewController isKindOfClass:[
       MenuViewController class]]) {
36         [self.slidingViewController setUnderLeftViewController:[self.
       storyboard instantiateViewControllerWithIdentifier:@"Menu"]];
37     }
38
39     [self.view addGestureRecognizer:self.slidingViewController.panGesture];
40
41     // Build navigation toolbar items
42     [self.navigationItem setLeftBarButtonItem:[NavigationToolbarBuilder
       createMenuBarButtonAtTarget:self]];
43     [self.navigationItem setTitleView:[NavigationToolbarBuilder
       createTitleBarButtonWithName:NSLocalizedString(@"ROUTES", nil)]];
```

```
44      UIButton *edit_button = [UIButton buttonWithType:UIButtonTypeCustom];
45      [edit_button setFrame:CGRectMake(0, 0, 30, 30)];
46      [edit_button setImage:[UIImage imageNamed:@"icon_edit.png"] forState:
        UIControlStateNormal];
47      [edit_button addTarget:self action:@selector(enterEditingMode:)
        forControlEvents:UIControlEventTouchUpInside];
48      UIBarButtonItem *edit_bar_button = [[UIBarButtonItem alloc]
        initWithCustomView:edit_button];
49      [self.navigationItem setRightBarButtonItem:edit_bar_button];
50  }
51
52  -(IBAction)openMenu:(id)sender {
53      [self.slidingViewController anchorTopViewTo:ECRight];
54  }
55
56  - (void)didReceiveMemoryWarning
57  {
58      [super didReceiveMemoryWarning];
59      // Dispose of any resources that can be recreated.
60  }
61
62  -(void)initData {
63      NSString *routes_query = @"SELECT id, name, date FROM track";
64      routes = [[NSMutableArray alloc]initWithArray:[[SQLiteManager singleton]
        executeSql:routes_query]];
65  }
66
67  -(void)deleteTrackAtRow:(NSInteger)row {
68      NSString *query_remove_track = [NSString stringWithFormat:@"DELETE FROM
        track WHERE id = %@", [[routes objectAtIndex:row]objectForKey:@"id"]];
69      [[SQLiteManager singleton]executeSql:query_remove_track];
70
71      NSString *query_remove_locations = [NSString stringWithFormat:@"DELETE
        FROM location WHERE track = %@", [[routes objectAtIndex:row]objectForKey:
        @"id"]];
72      [[SQLiteManager singleton]executeSql:query_remove_locations];
73  }
74
75  #pragma UITableView
76  - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(
        NSInteger)section {
77      // Return the number of rows in the section.
78      return [routes count];
79  }
80
81  - (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath
        :(NSIndexPath *)indexPath {
82      static NSString *CellIdentifier = @"Cell";
83      UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:
        CellIdentifier];
84      if (cell == nil) {
```

```
85          cell = [[UITableViewCell alloc] initWithStyle:
        UITableViewCellStyleSubtitle reuseIdentifier:CellIdentifier];
86      }
87
88      UIImageView *background_cell = [[UIImageView alloc]initWithImage:[UIImage
         imageNamed:@"resource_cell_background.png"]];
89      [background_cell setContentMode:UIViewContentModeScaleAspectFill];
90      [cell setBackgroundView:background_cell];
91
92      [cell.textLabel setText:[[routes objectAtIndex:indexPath.row]objectForKey
        :@"name"]];
93      [cell.textLabel setFont:[UIFont fontWithName:@"Euphemia UCAS" size
        :17.0]];
94      [cell.textLabel setTextColor:[UIColor colorWithRed:0.0/255.0 green
        :122.0/255.0 blue:255.0/255.0 alpha:1.0]];
95
96      [cell.detailTextLabel setText:[[routes objectAtIndex:indexPath.row]
        objectForKey:@"date"]];
97      [cell.detailTextLabel setFont:[UIFont fontWithName:@"Euphemia UCAS" size
        :11]];
98      // Set the accessory view.
99      UIImageView *imageView = [[UIImageView alloc] initWithImage:[UIImage
        imageNamed:@"icon_tableview.png"]];
100     [cell setAccessoryView:imageView];
101     [cell.accessoryView setFrame:CGRectMake(0, 0, 44, 44)];
102     return cell;
103 }
104
105 - (void)tableView:(UITableView *)tableView didSelectRowAtIndexPath:(
        NSIndexPath *)indexPath {
106     [tableView deselectRowAtIndexPath:indexPath animated:YES];
107     NSString *track_id = [[routes objectAtIndex:indexPath.row]objectForKey:@"
        id"];
108     NSString *identifier = @"TrackDetail";
109     TrackDetailViewController *track_detail = [self.storyboard
        instantiateViewControllerWithIdentifier:identifier];
110     [track_detail setTrack_id:track_id];
111     [track_detail setFrom_tracking:NO];
112     [self.navigationController pushViewController:track_detail animated:YES];
113 }
114
115 - (IBAction)enterEditingMode:(id)sender {
116     if ([table isEditing]) {
117         [table setEditing:NO animated:YES];
118         UIButton *edit_button = [UIButton buttonWithType:UIButtonTypeCustom];
119         [edit_button setFrame:CGRectMake(0, 0, 30, 30)];
120         [edit_button setImage:[UIImage imageNamed:@"icon_edit.png"] forState:
        UIControlStateNormal];
121         [edit_button addTarget:self action:@selector(enterEditingMode:)
        forControlEvents:UIControlEventTouchUpInside];
```

```
122         UIBarButtonItem *edit_bar_button = [[UIBarButtonItem alloc]
    initWithCustomView:edit_button];
123         [self.navigationItem setRightBarButtonItem:edit_bar_button];
124     }
125     else {
126         [table setEditing:YES animated:YES];
127         UIButton *edit_button = [UIButton buttonWithType:UIButtonTypeCustom];
128         [edit_button setFrame:CGRectMake(0, 0, 30, 30)];
129         [edit_button setImage:[UIImage imageNamed:@"icon_edit_done.png"]
    forState:UIControlStateNormal];
130         [edit_button addTarget:self action:@selector(enterEditingMode:)
    forControlEvents:UIControlEventTouchUpInside];
131         UIBarButtonItem *edit_bar_button = [[UIBarButtonItem alloc]
    initWithCustomView:edit_button];
132         [self.navigationItem setRightBarButtonItem:edit_bar_button];
133     }
134 }
135
136 - (void)tableView:(UITableView *)tableView commitEditingStyle:(
    UITableViewCellEditingStyle)editingStyle forRowAtIndexPath:(NSIndexPath
    *)indexPath {
137     if (editingStyle == UITableViewCellEditingStyleDelete) {
138         [self deleteTrackAtRow:indexPath.row];
139         [self initData];
140         [table reloadData];
141     }
142 }
143
144 -(void)viewDidAppear:(BOOL)animated {
145     [self initData];
146     [table reloadData];
147 }
148
149 @end
```

**MapViewController.m**

```
1  //
2  //  MapViewController.m
3  //  TrackMyRun
4  //
5  //  Created by David on 11/25/13.
6  //  Copyright (c) 2013 David. All rights reserved.
7  //
8
9  #import "MapViewController.h"
10 #import "NavigationToolbarBuilder.h"
11
12 @interface MapViewController ()
13
14 @end
```

```
15
16 @implementation MapViewController
17 @synthesize polyline;
18 @synthesize region;
19
20
21 - (void)viewDidLoad
22 {
23     [super viewDidLoad];
24    // Do any additional setup after loading the view.
25     [self.navigationItem setTitleView:[NavigationToolbarBuilder
       createTitleBarButtonWithName:NSLocalizedString(@"MAP", nil)]];
26     [self.navigationItem setLeftBarButtonItem:[NavigationToolbarBuilder
       createBackBarButtonWithTarget:self]];
27
28     [mapview setDelegate:self];
29     [mapview setRegion:region animated:YES];
30     [mapview addOverlay:polyline];
31 }
32
33 -(IBAction)goBack:(id)sender {
34     [self.navigationController popViewControllerAnimated:YES];
35 }
36
37 - (void)didReceiveMemoryWarning
38 {
39     [super didReceiveMemoryWarning];
40     // Dispose of any resources that can be recreated.
41 }
42
43 #pragma MKMapViewDelegate
44 - (MKOverlayView*)mapView:(MKMapView*)theMapView viewForOverlay:(id <
       MKOverlay>)overlay {
45     MKPolylineView* lineView = [[MKPolylineView alloc]initWithPolyline:self.
       polyline];
46     [lineView setFillColor:[UIColor whiteColor]];
47     [lineView setStrokeColor:[UIColor redColor]];
48     [lineView setLineWidth:4];
49
50     return lineView;
51 }
52
53 @end
```

**StatisticsViewController.m**

```
1 //
2 //  StatisticsViewController.m
3 //  TrackMyRun
4 //
5 //  Created by David on 10/22/13.
```

```
6  //  Copyright (c) 2013 David. All rights reserved.
7  //
8
9  #import "StatisticsViewController.h"
10 #import "ECSlidingViewController.h"
11 #import "MenuViewController.h"
12 #import "NavigationToolbarBuilder.h"
13 #import "CalculateStatistics.h"
14
15 @interface StatisticsViewController ()
16
17 @end
18
19 @implementation StatisticsViewController
20
21 - (void)viewDidLoad
22 {
23     [super viewDidLoad];
24    // Do any additional setup after loading the view.
25
26     // View properties
27     [self.view.layer setShadowOpacity:0.75f];
28     [self.view.layer setShadowRadius:10.0f];
29     [self.view.layer setShadowColor:[UIColor blackColor].CGColor];
30
31     if (![self.slidingViewController.underLeftViewController isKindOfClass:[
       MenuViewController class]]) {
32         [self.slidingViewController setUnderLeftViewController:[self.
       storyboard instantiateViewControllerWithIdentifier:@"Menu"]];
33     }
34
35     [self.view addGestureRecognizer:self.slidingViewController.panGesture];
36
37     // Build navigation toolbar items
38     [self.navigationItem setLeftBarButtonItem:[NavigationToolbarBuilder
       createMenuBarButtonAtTarget:self]];
39     [self.navigationItem setTitleView:[NavigationToolbarBuilder
       createTitleBarButtonWithName:NSLocalizedString(@"STATISTICS", nil)]];
40
41     [self calculateStatistics];
42     [scrollview setContentSize:CGSizeMake(self.view.frame.size.width, 577)];
43 }
44
45 -(IBAction)openMenu:(id)sender {
46     [self.slidingViewController anchorTopViewTo:ECRight];
47 }
48
49 - (void)didReceiveMemoryWarning
50 {
51     [super didReceiveMemoryWarning];
52     // Dispose of any resources that can be recreated.
```

```
53 }
54
55 -(void)calculateStatistics {
56     NSMutableDictionary *statistics = [CalculateStatistics
       getGlobalStatistics];
57     [label_number_tracks setText:NSLocalizedString(@"NUMBER_OF_TRACKS", nil)
       ];
58     [label_number_tracks setFont:[UIFont fontWithName:@"Euphemia UCAS" size
       :18.0]];
59
60     [label_duration setText:NSLocalizedString(@"DURATION_EMPTY", nil)];
61     [label_duration setFont:[UIFont fontWithName:@"Euphemia UCAS" size
       :18.0]];
62
63     [label_distance setText:NSLocalizedString(@"DISTANCE_EMPTY", nil)];
64     [label_distance setFont:[UIFont fontWithName:@"Euphemia UCAS" size
       :18.0]];
65
66     [label_mean_speed setText:NSLocalizedString(@"MEAN_SPEED_EMPTY", nil)];
67     [label_mean_speed setFont:[UIFont fontWithName:@"Euphemia UCAS" size
       :18.0]];
68
69     [label_max_speed setText:NSLocalizedString(@"MAX_SPEED_EMPTY", nil)];
70     [label_max_speed setFont:[UIFont fontWithName:@"Euphemia UCAS" size
       :18.0]];
71
72     [label_height_diff setText:NSLocalizedString(@"HEIGHT_DIFF_EMPTY", nil)];
73     [label_height_diff setFont:[UIFont fontWithName:@"Euphemia UCAS" size
       :18.0]];
74
75     [label_calories setText:NSLocalizedString(@"CALORIES_EMPTY", nil)];
76     [label_calories setFont:[UIFont fontWithName:@"Euphemia UCAS" size
       :18.0]];
77
78     NSString *number_tracks = [NSString stringWithFormat:@"%@ %@", [
       statistics objectForKey:@"tracks_number"], NSLocalizedString(@"TRACKS",
       nil)];
79     // Duration string.
80     int current_time = [[statistics objectForKey:@"duration"]floatValue];
81     int seconds = current_time % 60;
82     int minutes = (current_time / 60) % 60;
83     int hours = (current_time / 3600) % 60;
84     NSString *duration;
85     if (hours > 0) {
86         duration = [NSString stringWithFormat:@"%.2d:%.2d:%.2d %@", hours,
       minutes, seconds, NSLocalizedString(@"HOURS", nil)];
87     }
88     else {
89         duration = [NSString stringWithFormat:@"%.2d:%.2d %@", minutes,
       seconds, NSLocalizedString(@"MINUTES", nil)];
90     }
```

```
91      // Distance string.
92      float current_distance = [[statistics objectForKey:@"distance"]floatValue
        ];
93      NSString *distance;
94      if (current_distance >= 1000) {
95          current_distance = current_distance / 1000;
96          distance = [NSString stringWithFormat:@"%.2f km", current_distance];
97      }
98      else {
99          distance = [NSString stringWithFormat:@"%.0f %@", current_distance,
        NSLocalizedString(@"METERS", nil)];
100     }
101     NSString *mean_speed = [NSString stringWithFormat:@"%@ km/h", [statistics
         objectForKey:@"mean_speed"], NSLocalizedString(@"MEAN_SPEED", nil)];
102     NSString *max_speed = [NSString stringWithFormat:@"%@ km/h", [statistics
        objectForKey:@"max_speed"]];
103     NSString *height_diff = [NSString stringWithFormat:@"%@ m", [statistics
        objectForKey:@"height_diff"], NSLocalizedString(@"HEIGHT_DIFF", nil)];
104     NSString *calories = [NSString stringWithFormat:@"%@ kcal", [statistics
        objectForKey:@"calories"], NSLocalizedString(@"CALORIES", nil)];
105
106     [label_number_tracks_value setText:number_tracks];
107     [label_number_tracks_value setFont:[UIFont fontWithName:@"Euphemia UCAS"
        size:16.0]];
108
109     [label_duration_value setText:duration];
110     [label_duration_value setFont:[UIFont fontWithName:@"Euphemia UCAS" size
        :16.0]];
111
112     [label_distance_value setText:distance];
113     [label_distance_value setFont:[UIFont fontWithName:@"Euphemia UCAS" size
        :16.0]];
114
115     [label_mean_speed_value setText:mean_speed];
116     [label_mean_speed_value setFont:[UIFont fontWithName:@"Euphemia UCAS"
        size:16.0]];
117
118     [label_max_speed_value setText:max_speed];
119     [label_max_speed_value setFont:[UIFont fontWithName:@"Euphemia UCAS" size
        :16.0]];
120
121     [label_height_diff_value setText:height_diff];
122     [label_height_diff_value setFont:[UIFont fontWithName:@"Euphemia UCAS"
        size:16.0]];
123
124     [label_calories_value setText:calories];
125     [label_calories_value setFont:[UIFont fontWithName:@"Euphemia UCAS" size
        :16.0]];
126 }
127
128 @end
```

**AboutViewController.m**

```
1  //
2  //  SecondViewController.m
3  //  TrackMyRun
4  //
5  //  Created by David on 10/9/13.
6  //  Copyright (c) 2013 David. All rights reserved.
7  //
8
9  #import "AboutViewController.h"
10 #import "ECSlidingViewController.h"
11 #import "MenuViewController.h"
12 #import "NavigationToolbarBuilder.h"
13
14 @interface AboutViewController ()
15
16 @end
17
18 @implementation AboutViewController
19
20 - (void)viewDidLoad
21 {
22     [super viewDidLoad];
23    // Do any additional setup after loading the view.
24
25     // View properties
26     [self.view.layer setShadowOpacity:0.75f];
27     [self.view.layer setShadowRadius:10.0f];
28     [self.view.layer setShadowColor:[UIColor blackColor].CGColor];
29
30     if (![self.slidingViewController.underLeftViewController isKindOfClass:[
       MenuViewController class]]) {
31         [self.slidingViewController setUnderLeftViewController:[self.
       storyboard instantiateViewControllerWithIdentifier:@"Menu"]];
32     }
33
34     [self.view addGestureRecognizer:self.slidingViewController.panGesture];
35
36     // Build navigation toolbar items
37     [self.navigationItem setLeftBarButtonItem:[NavigationToolbarBuilder
       createMenuBarButtonAtTarget:self]];
38     [self.navigationItem setTitleView:[NavigationToolbarBuilder
       createTitleBarButtonWithName:NSLocalizedString(@"ABOUT", nil)]];
39
40     [self loadInterfaceElements];
41 }
42
43 -(IBAction)openMenu:(id)sender {
```

```
44      [self.slidingViewController anchorTopViewTo:ECRight];
45  }
46
47  -(void)loadInterfaceElements {
48      [label_name setText:NSLocalizedString(@"DEVELOPER_NAME", nil)];
49      [label_name setFont:[UIFont fontWithName:@"Euphemia UCAS" size:14.0]];
50      [label_name_value setText:@"David Montenegro Martinez"];
51      [label_name_value setFont:[UIFont fontWithName:@"Euphemia UCAS" size
        :16.0]];
52
53      [label_contact setText:NSLocalizedString(@"CONTACT_NAME", nil)];
54      [label_contact setFont:[UIFont fontWithName:@"Euphemia UCAS" size:14.0]];
55      [label_contact_value setText:@"davidmontenegro@uoc.edu"];
56      [label_contact_value setFont:[UIFont fontWithName:@"Euphemia UCAS" size
        :16.0]];
57
58      [label_web setText:NSLocalizedString(@"WEB_SUPPORT", nil)];
59      [label_web setFont:[UIFont fontWithName:@"Euphemia UCAS" size:14.0]];
60      [label_web_value setText:@"www.trackmyrun.com"];
61      [label_web_value setFont:[UIFont fontWithName:@"Euphemia UCAS" size
        :16.0]];
62
63      [label_version setText:[NSString stringWithFormat:@"%@: 1.29.",
        NSLocalizedString(@"VERSION", nil)]];
64      [label_version setFont:[UIFont fontWithName:@"Euphemia UCAS" size:12.0]];
65      [label_copyright setText:[NSString stringWithFormat:@"%@",
        NSLocalizedString(@"COPYRIGHT", nil)]];
66      [label_copyright setFont:[UIFont fontWithName:@"Euphemia UCAS" size
        :12.0]];
67  }
68
69  - (void)didReceiveMemoryWarning
70  {
71      [super didReceiveMemoryWarning];
72      // Dispose of any resources that can be recreated.
73  }
74
75  @end
```

**NavigationToolbarBuilder.m**

```
1   //
2   //  CustomToolbarHelper.m
3   //  TrackMyRun
4   //
5   //  Created by David on 10/22/13.
6   //  Copyright (c) 2013 David. All rights reserved.
7   //
8
9   #define SYSTEM_VERSION_GREATER_THAN_OR_EQUAL_TO(v)  ([[[UIDevice
        currentDevice] systemVersion] compare:v options:NSNumericSearch] !=
```

```
    NSOrderedAscending)
10
11  #import "NavigationToolbarBuilder.h"
12
13  @implementation NavigationToolbarBuilder
14
15  +(UIToolbar*)createCustomToolbarInViewController:(UIViewController*)
        viewController{
16      // Toolbar properties for iOS 6 & iOS 7, unused
17      UIToolbar *toolbar = [[UIToolbar alloc]init];
18      [toolbar setClipsToBounds:YES];
19      [toolbar setTranslucent:NO];
20      if (SYSTEM_VERSION_GREATER_THAN_OR_EQUAL_TO(@"7.0")) {
21          [toolbar setFrame:CGRectMake(0, 20, viewController.view.bounds.size.
        width, 44)];
22      }
23      else {
24          [toolbar setFrame:CGRectMake(0, 0, viewController.view.bounds.size.
        width, 44)];
25          [toolbar setTintColor:[UIColor clearColor]];
26      }
27
28      return toolbar;
29  }
30
31  +(UIBarButtonItem*)createMenuBarButtonAtTarget:(UIViewController*)parentView
        {
32      // Custom menu bar button with a parentview target
33      UIButton *menu_button = [UIButton buttonWithType:UIButtonTypeCustom];
34      [menu_button setFrame:CGRectMake(0, 0, 30, 30)];
35      [menu_button setImage:[UIImage imageNamed:@"button_menu.png"] forState:
        UIControlStateNormal];
36      [menu_button addTarget:parentView action:@selector(openMenu:)
        forControlEvents:UIControlEventTouchUpInside];
37      UIBarButtonItem *menu_bar_button = [[UIBarButtonItem alloc]
        initWithCustomView:menu_button];
38
39      return menu_bar_button;
40  }
41
42  +(UIBarButtonItem*)createBackBarButtonWithTarget:(UIViewController*)
        parentView {
43      // Custom back bar button with a parentview target
44      UIButton *back_button = [UIButton buttonWithType:UIButtonTypeCustom];
45      [back_button setFrame:CGRectMake(0, 0, 30, 30)];
46      [back_button setImage:[UIImage imageNamed:@"button_back.png"] forState:
        UIControlStateNormal];
47      [back_button addTarget:parentView action:@selector(goBack:)
        forControlEvents:UIControlEventTouchUpInside];
48      UIBarButtonItem *back_bar_button = [[UIBarButtonItem alloc]
        initWithCustomView:back_button];
```

```
49
50      return back_bar_button;
51  }
52
53  +(UIView*)createTitleBarButtonWithName:(NSString*)title{
54      // Custom title bar
55      UILabel *title_label = [[UILabel alloc]initWithFrame:CGRectMake(0, 0,
        130, 44)];
56      [title_label setTextAlignment:NSTextAlignmentCenter];
57      [title_label setText:title];
58      [title_label setFont:[UIFont fontWithName:@"Euphemia UCAS" size:22]];
59      [title_label setBackgroundColor:[UIColor clearColor]];
60      [title_label setTextColor:[UIColor colorWithRed:0.0/255.0 green
        :122.0/255.0 blue:255.0/255.0 alpha:1.0]];
61
62      return (UIView*)title_label;
63  }
64
65  @end
```

**LocationService.m**

```
1   //
2   //  LocationService.m
3   //  TrackMyRun
4   //
5   //  Created by David on 11/21/13.
6   //  Copyright (c) 2013 David. All rights reserved.
7   //
8
9   #import "LocationService.h"
10  #import "CalculateStatistics.h"
11
12  @implementation LocationService
13
14  @synthesize delegate;
15  @synthesize track;
16  @synthesize duration;
17
18  -(void)start {
19      // Init object data.
20      last_location = nil;
21      counter = 0;
22      total_distance = 0;
23      // Init the track with the default values.
24      [self createTrack];
25      // Init the location manager with custom properties.
26      location_manager = [[CLLocationManager alloc]init];
27      [location_manager setDelegate:self];
28      [location_manager setDesiredAccuracy:kCLLocationAccuracyBest];
29      [location_manager setDistanceFilter:1];
```

```
30      [location_manager startUpdatingLocation];
31  }
32
33  -(BOOL)stop {
34      BOOL valid = NO;
35      // Stop hearing Location Manager update location.
36      [location_manager stopUpdatingLocation];
37      if (last_location != nil && total_distance > 25) {
38          valid = YES;
39          // Save track properties.
40          [CalculateStatistics calculateStatisticsOfTrackId:track.identifier
        withDuration:duration andDistance:total_distance];
41      }
42      else{
43          // Delete the track.
44          [track deleteTrack];
45          UIAlertView *alert = [[UIAlertView alloc] initWithTitle:
        NSLocalizedString(@"ALERT", nil) message:NSLocalizedString(@"
        NO_POINTS_RECIEVED", nil) delegate:self cancelButtonTitle:nil
        otherButtonTitles:@"OK", nil, nil];
46          [alert show];
47      }
48      return valid;
49  }
50
51  -(void)createTrack {
52      track = [[Track alloc]init];
53      [track initialize];
54      [track save];
55  }
56
57
58  #pragma CLLocationManager
59  - (void)locationManager:(CLLocationManager*)manager didUpdateToLocation:(
        CLLocation*)newLocation fromLocation:(CLLocation*)oldLocation {
60      // Get the new recieved location.
61      Location *new_location = [[Location alloc]init];
62      [new_location setData:newLocation];
63      [new_location setRecievedProperties];
64      // Validate the new location with her own properties.
65      BOOL valid = [new_location validate];
66      if (valid) {
67          // If the last location is null/nil, save the new one.
68          if (last_location == nil) {
69              last_location = new_location;
70          }
71          // Else, check the properties between them.
72          else if(counter < 3) {
73              valid = [new_location validatePropertiesWithPreviousLocation:
        last_location];
74          }
```

```
75          else if(counter == 3) {
76              valid = YES;
77          }
78          // If still valid.
79          if (valid) {
80              // Insert location to database.
81              [new_location saveWithTrackId:track.identifier];
82              last_location = new_location;
83              counter = 0;
84
85              total_distance = total_distance + new_location.
    distance_with_previous;
86              NSString *distance = [NSString stringWithFormat:@"%ld",
    total_distance];
87
88              float total_calories = [CalculateStatistics
    getCaloriesForDistance:total_distance];
89              NSString *calories = [NSString stringWithFormat:@"%.0f",
    total_calories];
90
91              NSMutableDictionary *userInfo = [[NSMutableDictionary alloc]
    initWithObjects:[NSArray arrayWithObjects:new_location.speed, distance,
    calories, nil] forKeys:[NSArray arrayWithObjects:@"speed", @"distance", @
    "calories", nil]];
92              [[NSNotificationCenter defaultCenter] postNotificationName: @"
    tracking" object:nil userInfo:userInfo];
93
94          }
95          else {
96              counter++;
97          }
98      }
99  }
100
101 @end
```

**Track.m**

```
1  //
2  //  Track.m
3  //  TrackMyRun
4  //
5  //  Created by David on 11/21/13.
6  //  Copyright (c) 2013 David. All rights reserved.
7  //
8
9  #import "Track.h"
10 #import "SQLiteManager.h"
11
12 @implementation Track
13
```

```
14 @synthesize identifier;
15
16 -(void)initialize {
17     NSDateFormatter *formatter = [[NSDateFormatter alloc] init];
18     [formatter setDateFormat:@"dd/MM/yyyy"];
19     NSString *current_date = [formatter stringFromDate:[NSDate date]];
20
21     identifier = nil;
22     name = NSLocalizedString(@"UNSAVED", nil);
23     date = current_date;
24     duration = @"0";
25     distance = @"0";
26     max_speed = @"0";
27     mean_speed = @"0";
28     height_diff = @"0";
29     calories = @"0";
30 }
31
32 -(void)save {
33     // Save the current data.
34     NSMutableDictionary *track_data = [NSMutableDictionary
       dictionaryWithObjectsAndKeys:name, @"name", date, @"date", duration, @"
       duration", distance, @"distance", max_speed, @"max_speed", mean_speed, @"
       mean_speed", height_diff, @"height_diff", calories, @"calories", nil];
35     [[SQLiteManager singleton]save:track_data into:@"track"];
36     // Get the track identifier
37     NSString *user_query = @"SELECT id FROM track ORDER BY ID DESC LIMIT 1";
38     NSArray *track_id = [[SQLiteManager singleton]executeSql:user_query];
39     identifier = [[track_id objectAtIndex:0]objectForKey:@"id"];
40 }
41
42 -(void)deleteTrack {
43     NSString *query_remove_track = [NSString stringWithFormat:@"DELETE FROM
       track WHERE id = %@", identifier];
44     [[SQLiteManager singleton]executeSql:query_remove_track];
45 }
46
47
48
49 @end
```

**Location.m**

```
1 //
2 //  Location.m
3 //  TrackMyRun
4 //
5 //  Created by David on 11/21/13.
6 //  Copyright (c) 2013 David. All rights reserved.
7 //
8
```

```
9  #import "Location.h"
10 #import "SQLiteManager.h"
11
12 @implementation Location
13 @synthesize data;
14 @synthesize speed;
15 @synthesize distance_with_previous;
16
17 -(BOOL)validate{
18     NSLog(@"- LOCATION RECIEVED -");
19     NSLog(@"Latitude: %f", data.coordinate.latitude);
20     NSLog(@"Longitude: %f", data.coordinate.latitude);
21     NSLog(@"Accuracy: %f", data.horizontalAccuracy);
22     NSLog(@"Speed: %f", data.speed);
23     NSLog(@"-");
24
25     valid = YES;
26     // If the coordinates are wrong.
27     if (data.coordinate.latitude == 0.0f || data.coordinate.longitude == 0.0f
       ) {
28         valid = NO;
29     }
30     // If the accuracy is more than the limit
31     if (data.horizontalAccuracy > 40.1f) {
32         valid = NO;
33     }
34     // If the speed is more than 10m/s (36km/h).
35     if (data.speed > 10.0f && data.speed < 0.0f) {
36         valid = NO;
37     }
38
39     return valid;
40 }
41
42 -(void)setRecievedProperties {
43     // Set recieved properties from the devie GPS.
44     latitude = [[NSString alloc]initWithFormat:@"%.8f", data.coordinate.
       latitude];
45     longitude = [[NSString alloc]initWithFormat:@"%.8f", data.coordinate.
       longitude];
46     altitude = [[NSString alloc]initWithFormat:@"%.0f", data.altitude];
47     double current_timestamp = [data.timestamp timeIntervalSince1970];
48     timestamp = [[NSString alloc]initWithFormat:@"%f", current_timestamp];
49     accuracy = [[NSString alloc]initWithFormat:@"%.0f", data.
       horizontalAccuracy];
50     speed = [[NSString alloc]initWithFormat:@"%.2f", data.speed*3.6];
51 }
52
53 -(BOOL)validatePropertiesWithPreviousLocation:(Location*)last_location{
54     double distance = [self.data distanceFromLocation:last_location.data];
55     if (distance > 31.01f) {
```

```
56          return NO;
57      }
58
59      double time = self.data.timestamp.timeIntervalSince1970 - last_location.
        data.timestamp.timeIntervalSince1970;
60      double velocity = (distance/time)*3.6;
61      if (velocity > 35) {
62          return NO;
63      }
64      distance_with_previous = distance;
65      speed = [[NSString alloc]initWithFormat:@"%.2f", velocity];
66
67      return YES;
68  }
69
70
71  -(void)saveWithTrackId:(NSString*)track_id {
72      NSMutableDictionary *track_data = [NSMutableDictionary
        dictionaryWithObjectsAndKeys:latitude, @"latitude", longitude, @"
        longitude", altitude, @"altitude", timestamp, @"timestamp", accuracy, @"
        accuracy", speed, @"speed", track_id, @"track", nil];
73      [[SQLiteManager singleton]save:track_data into:@"location"];
74  }
75
76  @end
```

**CalculateStatistics.m**

```
1   //
2   //  CalculateStatistics.m
3   //  TrackMyRun
4   //
5   //  Created by David on 11/24/13.
6   //  Copyright (c) 2013 David. All rights reserved.
7   //
8
9   const float kCalFact = 0.1036;
10
11  #import "CalculateStatistics.h"
12  #import "SQLiteManager.h"
13
14  @implementation CalculateStatistics
15
16  +(void)calculateStatisticsOfTrackId:(NSString*)identifier withDuration:(float
        )duration andDistance:(float)distance {
17      // Get the track data.
18      NSString *track_query = [NSString stringWithFormat:@"SELECT name, date
        FROM track WHERE id = '%@'", identifier];
19      NSArray *track_result = [[NSArray alloc]initWithArray:[[SQLiteManager
        singleton]executeSql:track_query]];
```

```
20      NSMutableDictionary *track_data = [[NSMutableDictionary alloc]
        initWithDictionary:[track_result objectAtIndex:0]];
21
22      // Get the location rows from database.
23      NSString *location_query = [NSString stringWithFormat:@"SELECT * FROM
        location WHERE track = '%@'", identifier];
24      NSArray *location_result = [[NSArray alloc]initWithArray:[[SQLiteManager
        singleton]executeSql:location_query]];
25
26      float max_speed = 0;
27      float mean_speed = 0;
28      float height_diff = 0;
29      float min_height = 0;
30      float max_height = 0;
31      float total_speed = 0;
32      float number_of_locations = 0;
33
34      for (NSDictionary *location in location_result) {
35          // Check the speed.
36          float location_speed = [[location objectForKey:@"speed"]floatValue];
37          if (location_speed > max_speed) {
38              max_speed = location_speed;
39          }
40          total_speed = total_speed + location_speed;
41
42          // Check the altitude
43          float location_altitude = [[location objectForKey:@"altitude"]
        floatValue];
44          if (location_altitude < min_height) {
45              min_height = location_altitude;
46          }
47          else if (location_altitude > max_height) {
48              max_height = location_altitude;
49          }
50          number_of_locations++;
51      }
52      mean_speed = total_speed / number_of_locations;
53      height_diff = max_height - min_height;
54
55      float calories = [self getCaloriesForDistance:distance];
56
57      NSString *duration_string = [NSString stringWithFormat:@"%.0f", duration
        ];
58      NSString *distance_string = [NSString stringWithFormat:@"%.0f", distance
        ];
59      NSString *max_speed_string = [NSString stringWithFormat:@"%.2f",
        max_speed];
60      NSString *mean_speed_string = [NSString stringWithFormat:@"%.2f",
        mean_speed];
61      NSString *height_diff_string = [NSString stringWithFormat:@"%.0f",
        height_diff];
```

```
62      NSString *calories_string = [NSString stringWithFormat:@"%.1f", calories
        ];
63
64      // Save the current data.
65      NSMutableDictionary *track_update = [NSMutableDictionary
        dictionaryWithObjectsAndKeys:identifier, @"id", [track_data objectForKey:
        @"name"], @"name", [track_data objectForKey:@"date"], @"date",
        duration_string, @"duration", distance_string, @"distance",
        max_speed_string, @"max_speed", mean_speed_string, @"mean_speed",
        height_diff_string, @"height_diff", calories_string, @"calories", nil];
66      [[SQLiteManager singleton]save:track_update into:@"track"];
67  }
68
69  +(float)getCaloriesForDistance:(float)distance {
70      // Get the user data.
71      NSString *user_query = [NSString stringWithFormat:@"SELECT * FROM user"];
72      NSArray *user_result = [[NSArray alloc]initWithArray:[[SQLiteManager
        singleton]executeSql:user_query]];
73      NSMutableDictionary *user_data = [[NSMutableDictionary alloc]
        initWithDictionary:[user_result objectAtIndex:0]];
74
75      // Default weight
76      int weight = 80;
77      if (![[user_data objectForKey:@"weight"]isEqualToString:@""]) {
78          weight  = [[user_data objectForKey:@"weight"]intValue];
79      }
80
81      if ([[user_data objectForKey:@"sex"]isEqualToString:@"Male"]) {
82          return weight*kCalFact*0.01*distance;
83      }
84      else {
85          return weight*(kCalFact*0.9)*0.01*distance;
86      }
87  }
88
89  +(NSMutableDictionary*)getGlobalStatistics {
90      // Get the user data.
91      NSString *track_query = [NSString stringWithFormat:@"SELECT * FROM track"
        ];
92      NSArray *track_result = [[NSArray alloc]initWithArray:[[SQLiteManager
        singleton]executeSql:track_query]];
93
94      NSMutableDictionary *statistics = [[NSMutableDictionary alloc]init];
95
96      float duration = 0;
97      float distance = 0;
98      float max_speed = 0;
99      float mean_speed = 0;
100     float height_diff = 0;
101     float calories = 0;
102
```

```
103     for (NSDictionary* track in track_result) {
104         float duration_temp = [[track objectForKey:@"duration"]floatValue];
105         duration = duration + duration_temp;
106 
107         float distance_temp = [[track objectForKey:@"distance"]floatValue];
108         distance = distance + distance_temp;
109 
110         float max_speed_temp = [[track objectForKey:@"max_speed"]floatValue];
111         if (max_speed_temp > max_speed) {
112             max_speed = max_speed_temp;
113         }
114 
115         float mean_speed_temp = [[track objectForKey:@"mean_speed"]floatValue
    ];
116         mean_speed = mean_speed + mean_speed_temp;
117 
118         float height_diff_temp = [[track objectForKey:@"height_diff"]
    floatValue];
119         height_diff = height_diff + height_diff_temp;
120 
121         float calories_temp = [[track objectForKey:@"calories"]floatValue];
122         calories = calories + calories_temp;
123     }
124     if (track_result.count > 0) {
125         mean_speed = mean_speed / [track_result count];
126     }
127 
128     NSString *tracks_number = [NSString stringWithFormat:@"%lu", (unsigned
    long)track_result.count];
129     [statistics setObject:tracks_number forKey:@"tracks_number"];
130 
131     NSString *duration_string = [NSString stringWithFormat:@"%.2f", duration
    ];
132     [statistics setObject:duration_string forKey:@"duration"];
133 
134     NSString *distance_string = [NSString stringWithFormat:@"%.0f", distance
    ];
135     [statistics setObject:distance_string forKey:@"distance"];
136 
137     NSString *max_speed_string = [NSString stringWithFormat:@"%.0f",
    max_speed];
138     [statistics setObject:max_speed_string forKey:@"max_speed"];
139 
140     NSString *mean_speed_string = [NSString stringWithFormat:@"%.2f",
    mean_speed];
141     [statistics setObject:mean_speed_string forKey:@"mean_speed"];
142 
143     NSString *height_diff_string = [NSString stringWithFormat:@"%.0f",
    height_diff];
144     [statistics setObject:height_diff_string forKey:@"height_diff"];
145 
```

```
146     NSString *calories_string = [NSString stringWithFormat:@"%.1f", calories
        ];
147     [statistics setObject:calories_string forKey:@"calories"];
148 
149     return statistics;
150 }
151 
152 @end
```

# 15 Annexos

Aquesta secció descriu les parts del projectes no fonamentals, però que cal conèixer per saber les plataformes utilitzades, els respectius llenguatges, la seva sintàxis i per tant, la seva dificultat.

## 15.1 Xcode

S'ha explicat què és la plataforma *Xcode* i com funciona, però si es vol entrar en més detall, s'explica quines són les eines i les propietats que el fan un entorn on és molt còmode treballar-hi.

### 15.1.1 Vista de finestra única

Una de les propietats més característiques d'*Xcode*, és la nombrosa quantitat de finestres que s'utilitzen per realitzar les tasques de desenvolupament, s'han consolidat en una sola (Single-View). L'àrea d'*Xcode* treballa varis elements de la interfície d'usuari que permeten que sigui fàcil trebllar en diferents tàsques, fins i tot diferents projectes, sense aturar l'àrea de treball. Sempre està situat a l'àrea central de l'editor.

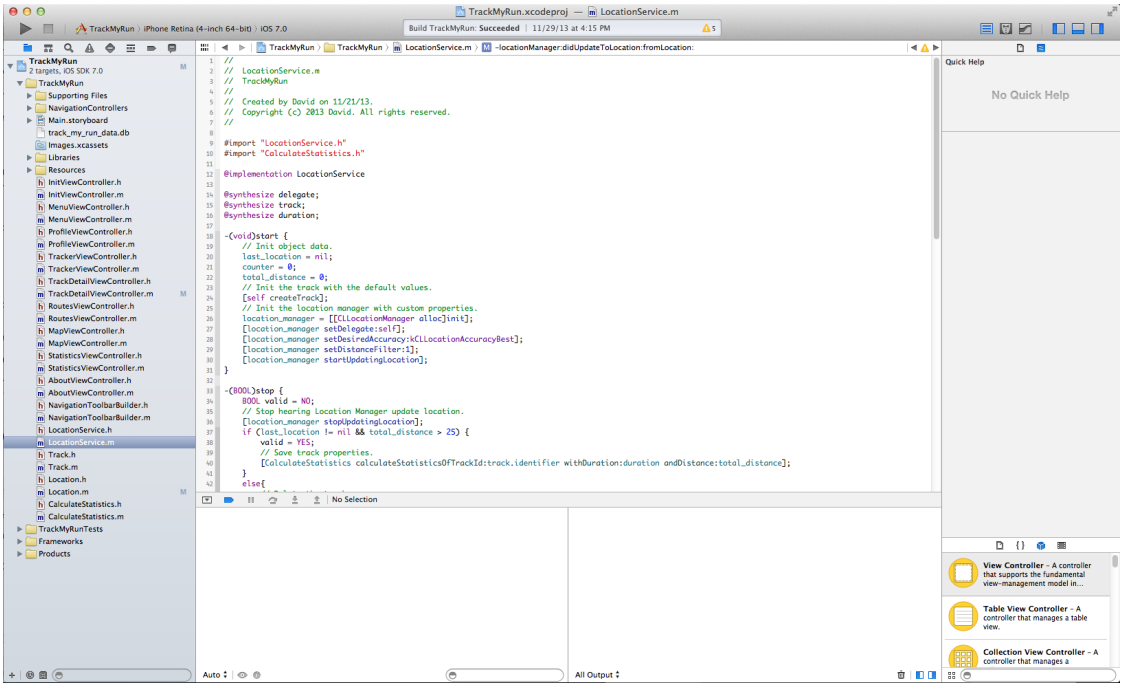

**Figura 29:** Vista de finestra única d'Xcode

A més, l'entorn també conté un navegador a l'esquerre de la vista, i inclou una llista dels fitxers del projecte, els símbols ordenats, una interfície de busca centralitzada, seguiment de problemes, dades de depuració amb seguiments de pila, punts de ruptura actius i inactius i una col·lecció permanent dels registres. El navegador de la interfície d'usuari

unificada ofereix el filtrat de contingut i els resultats de la cerca, de manera que el programador pot centrar-se amb la tasca actual.

A la part superior de cada panell de l'editor hi ha una barra que mostra la ubicació relativa a l'arxiu actual. Si es prem a qualsevol lloc de la ruta pot saltar immediatament a qualsevol altre fitxer del nivell.

### 15.1.2 Disseny d'interfícies

En versions anteriors, l'editor d'interfícies *Interface Builder* duia a terme el disseny d'interfícies en una vista separada, però actualment està tot a la mateixa. Seleccionant un fitxer d'interfície o l'*storyboard*, l'*Xcode* permet editar el contingut/disseny de la vista seleccionada. Obrint la component dreta de la vista, es mostra la gamma completa d'elements d'interfície, així com la biblioteca de controls i objectes de la interfície d'usuari. Es pot arrossegar un control de la biblioteca cap a la vista, i aquesta l'afegeix automàticament.

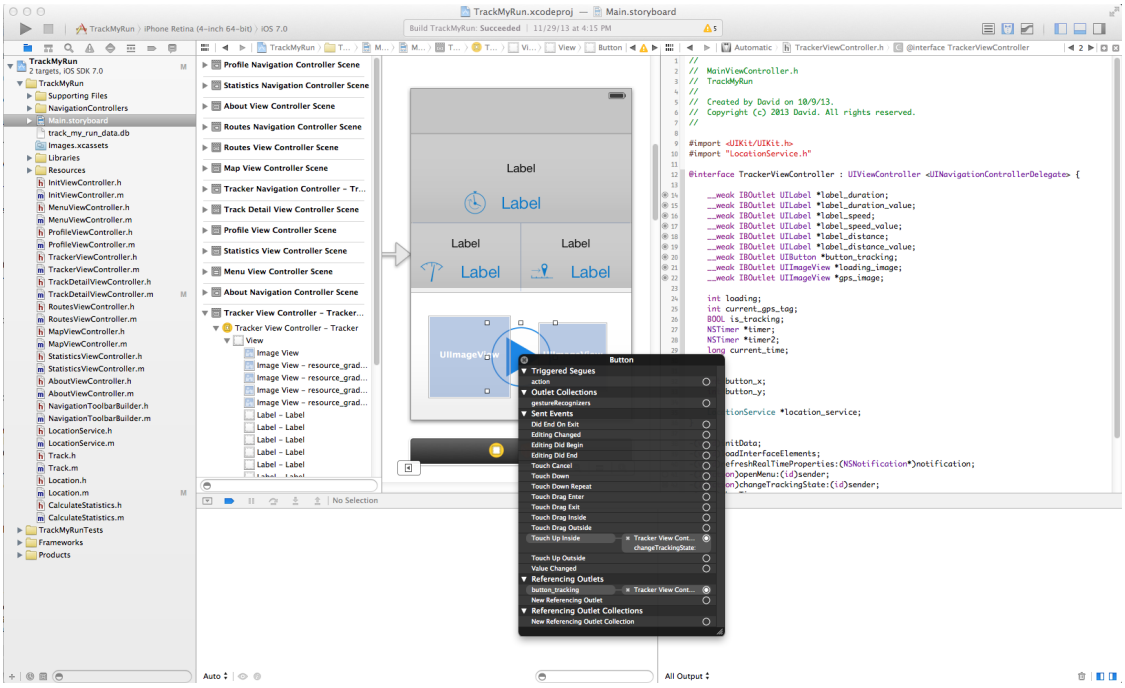


**Figura 30:** *Interface Builder* d'Xcode

Un dels seus punts més forts, és que es pot connectar directament de la vista de disseny al codi corresponent a la vista. Pot crear accions i sortides només arrastrant la connexió ja existent. S'ha de reconèixer que és una funcionalitat molt pràctica.

### 15.1.3 Compilador LLVM

LLVM 5 d'*Apple* és més ràpid i compila el codi el doble de ràpid que GCC, genera aplicacions que s'executen més ràpidament. El compilador va ser construït des d'un principi com un conjunt de llibreries altament optimitzades, fàcils d'estendre, fàcils d'optimitzar i dissenyades per les arquitectures dels últims xips. El precompilador de la pila LLVM té gran suport per a C, C++ i *Objective-C*. El ressaltat de la sintaxi, el codi complet i totes les característiques estan basades per l'analitzador d'LLVM.

### 15.1.4 Editor de versions

Una eina fonamental pel desenvolupament del projecte ha estat l'*SmartGit*: cada vegada que es realitzava un avenç en el projecte es guardava el seu estat, per si de cas en un determinat moment del desenvolupament, es volia tornar a un estat anterior o es volien veure els fitxer que havien canviat. S'ha executat de forma local, sense sincronitzar a cap repositori remot (normalment orientat al treball en grup).

**Figura 31:** Logo de l'*SmartGit*, l'editor de versions

## 15.2 LaTeX

TeX és un programa d'ordinador de *Donald E. Knuth*. Està orientat a la composició i impressió de fragments de text i fòrmules matemàtiques. LaTeX és un paquet de macros que permet a l'autor d'un text compondre i imprimir el document amb la major qualitat tipogràfica, utilitzant en conseqüència una sèrie de patrons prèviament definits.

### 15.2.1 Disseny de LaTeX

Normalment, per una publicació, un dissenyador de llibres intenta encertar les intencions de l'autor mentre ha realitzat l'escrit. Llavors, decideix de quina manera presentar-ho, com presentar els títols, cites, exemples, fòrmules, etc, bastant-se en el seu saber professional i sobre el contingut de l'escrit. Dins un entorn LaTeX, LaTeX és el dissenyador, per tant ha de rebre informació proporcionada per l'autor, a través d'instruccions o ordres. És bastant diferent a la majoria de processadors de text tals com *Microsoft Word* o *WordPerfect*. Amb aquestes aplicacions l'autor estableix el format del text amb l'entrada interactiva al introduir-ho a l'ordinador. De totes maneres, existeixen eines que permeten mostrar per pantalla el que finalment s'obté d'haver processat els fitxers amb LaTeX.

**Figura 32:** Logo de LaTeX

El disseny tipogràfic és una artesania que s'ha d'aprendre. Els autors inexperts cometen freqüentment errors de disseny. Molts creuen que un error en el disseny tipogràfic és una qüestió d'estètica: si el document presenta un bon aspecte des del punt de vista artístic, llavors està ben dissenyat. Malgrat això, els documents cal llegir-los, no penjar-los a un museu, doncs és més important una major **lectura** i **comprensió** que un aspecte agradable.

- S'ha d'escollir el tamany de les lletres i la numeració dels títols de manera que l'estructura dels capítols i les seccions sigui fàcilment identificable.
- S'ha d'escollir la longitud dels espais de mode que s'eviti el moviment fatigós dels ulls del lector i no perquè omplin, a ser possible, les pàgines amb un aspecte estèticament bo.

### 15.2.2 Avantatges i iconvenients de LaTeX

Quan es comença un debat com aquest, pot haver-hi molts punts de vista, però sempre s'ha de seguir el més objectiu. Alguns de les avantatges són:

- Existeix una major qualitat en els dissenys de text professionals a disposició, amb els que realment es poden crear documents com si fòssin d'imprenta.
- Es facilita la composició de fòrmules.
- L'usuari només necessita introduir instruccions senzilles amb les que s'indica l'estructura del document. Quasi mai cal preocupar-se pels detalls de creació amb tècniques d'impressió.
- Existeixen paquets addicionals sense cost per moltes tasques tipogràfiques que no es faciliten directament al LaTeX bàsic.
- Els autors tenen tendència a escriure textos ben estructurats perquè així és com treballa LaTeX. TeX, la màquina de composició, és portable i gratis. Funciona pràcticament a qualsevol plataforma.

Naturalment, també té inconvenients:

- Requereix cert temps per l'aprenentatge del llenguatge. És un procés no prescindible que cal assolir per obtenir una documentació com la desitjada inicialment.
- Per fer funcionar un sistema LaTeX es necessiten més recursos (memòria, espai de disc i potència de processat) que un processador de textos simple. Si s'analitza l'ús del processador, es pot veure que LaTeX supera la utilització de la CPU però només en el moment de compilació. Per contra, els altres l'utilitzen continuament.
- Es poden ajustar paràmetres de dissenys de document predefinits, però la creació de nous dissenys és molt difícil i requereix molt de temps.

## 15.3 Mètodes de distribució

Les aplicacions d'iOS es poden alliberar de diverses maneres. La més comuna (únic mètode d'obtenció per un usuari estàndard), és mitjançant l'*App Store*. Els desenvolupadors penjen les seves aplicacion al repositori d'*Apple* i un cop validades, es posen a disposició del gran públic. Existeix un altre mètode (normalment usat per alliberar versions prèvies a usuaris específics: clients, avaluadors, ...) que s'anomena *Ad Hoc*. Consisteix a distribuir una aplicació generant un certificat única i exclusivament per un nombre determinat de dispositius. A continuació es descriuen els passos que cal seguir per cada metodologia.

### 15.3.1 Release App Store

Al tractar-se d'una aplicació per a mòbils amb sistema operatiu iOS, no suposa un camí fàcil a l'hora de treure l'aplicació a la llum gràcies a la política d'admissió d'aplicacions i els restrictius acords de confidencialitat imposats per *Apple*. De totes maneres, tot desenvolupador ha d'assolir el seu objectiu, és a dir, veure la seva aplicació publicada a l'*App Store*. Els diferents passos que s'han de seguir són:

- Inscripció a *iOS Development Center* i descàrrega de l'SDK. Un cop dut a terme aquest pas, es poden començar a desenvolupar les aplicacions, compilar-les, i provar-les amb el simulador.
- Inscripció al programa per a desenvolupadors. El cost total de la llicència és de 99 dòlars, i s'exerceix el dret a poder executar aplicacions en un dispositiu propi.
- Certificats: Per defecte, un *iPhone* només té dret a executar aplicacions firmades per la casa, per tant és impescindible crear un perfil amb l'identificador del propi telèfon (mitjançant *Xcode*) introduint un altre formulari a la web. Amb aquesta sol·licitud s'obté un certificat que permet al desenvolupador instal·lar l'aplicació al mòbil associat.
- Distribució: El primer que cal fer és obtenir un altre certificat especial de distribució de tipus *App Store*, que s'ha de sol·licitar seguint el mateix procés que el certificat de desenvolupament. Un cop obtingut el certificat en qüestió, s'ha d'adaptar el nou perfil de distribució a *Xcode*.
- Enviament a *Apple*. Sobretot, cal tenir en compte que les opcions dels formularis web han de complir estrictament tots els seus requisits. Referent a les dades de l'aplicació i les funcionalitats natives que s'utilitzin, si no s'especifiquen cuidadosament a la descripció de l'aplicació, serà rebutjada.
- Revisió d'*Apple*: Un cop enviada l'aplicació, passa a la cua de revisió. El temps estimat d'acceptació és d'aproximadament cinc o sis dies laborals. Un cop passat aques temps, l'estat de l'aplicació passa a ser acceptada o rebutjada. En cas que no hagi estat acceptada, s'adjunta una nota que especifica la raó per la qual no s'ha acceptat. Amb la nota proporcionada, el programador ha de ser capaç de solucionar els inconvenients imposats. En cas que sigui acceptada, es pot escollir la seva data de publicació. Aquest pas acostuma a ser el més costós de tot el procés.
- Acceptació a l'*App Store*: Un cop acceptada, el desenvolupador pot posar-la a disposició del gran públic quan li sigui convenient. Un cop arribats a aquest punt, es pot donar el procés com a acabat.

És llavors quan, mitjançant *iTunes*, se n'obté un *feedback*. Cal tenir en copte l'opinió dels usuaris, i és necessari tenir-los satisfets.

### 15.3.2 Release Ad Hoc

A diferència del mètode de distribució anterior, no cal esperar a que validin l'aplicació, sinó que al ser per usuaris limitats, en cas que contingui errors o infringeixi qualsevol de les lleis imposades, és problema del desenvolupador. Els passos a seguir doncs, són els següents:

- La inscripció a *iOS Development Center*, i la inscripció al programa per a desenvolupadors són passos també indispensables, així que és el mateix procés que el mètode anterior.
- Donar d'alta l'identificador dels dispositius de destí: Cal identificar els dispositius als que va dirigida l'aplicació que es vol compartir, així que mitjançant *iTunes* s'obté l'identificador únic del dispositiu, i es dona d'alta al portal de desenvolupadors. El límit de dispositius a registrar és de 100.
- Certificat de distribució: Aquest cop el certificat ha de ser de tipus *Ad Hoc*. Cal sol·licitar-lo i facilitar dades tals com: l'identificador de l'aplicació que es vol distribuir (prèviament donada d'alta) i els dispositius als quals va dirigida.
- Generar executable: L'últim pas és generar l'executable, amb els certificats indicats i totes les propietats de l'aplicació en ordre.

Un cop l'aplicació arriba al destinatari, a través de l'*iTunes* s'està habilitat per instal·larla al dispositiu.